

# **Data-Driven Methods to Build Robust Legged Robots**

by

George Council

A dissertation submitted in partial fulfillment  
of the requirements for the degree of  
Doctor of Philosophy  
(Electrical Engineering:Systems)  
in The University of Michigan  
2020

Doctoral Committee:

Associate Professor Shai Revzen, Chair  
Associate Professor Dmitry Berenson  
Professor Jessy Grizzle  
Professor Ian Hiskens  
Assistant Professor Elliott Rouse

George Council

[gcouncil@umich.edu](mailto:gcouncil@umich.edu)

ORCID iD: 0000-0001-7666-8793

© George Council 2020

## ACKNOWLEDGEMENTS

I'd like to acknowledge and thank the essential role my adviser, Shai Revzen, had in fostering my education. Without his rather unconventional recruitment strategy, the intellectual growth I've experienced under his tutelage would've never occurred. His ability to wrangle my eclectic interests into a cohesive research direction was an impressive feat in its own right, and for that I'm grateful. I'd also like to thank Matthew Kvalheim, for his enthusiasm towards abstract mathematics proved to infectious. His willingness to laboriously explain theorems until I understood them sparked an unexpected, but thoroughly enjoyed, appreciation of mathematics. I'd furthermore like to acknowledge the roll Brian Bittner played in the converse direction. He often demanded to know if the mathematical rambling I was engaged in had any practical value, which certainly forced me to appreciate the difference between abstract nonsense and the practically useful. I'd like to also thank all the other members of BIRDS Lab - the broad collection of undergraduate and graduate students provided a stimulating and productive environment; through the piles of burnt circuitry, broken modules, and overheating batteries, it was a wild ride bringing theory to practice. Finally, I'd like to thank my wife, Char. She believed in me more than I did, and this document wouldn't exist without her extraordinary support. I love you.

# TABLE OF CONTENTS

<b>ACKNOWLEDGEMENTS</b> . . . . .	ii
<b>LIST OF FIGURES</b> . . . . .	vi
<b>LIST OF TABLES</b> . . . . .	xii
<b>LIST OF APPENDICES</b> . . . . .	xiii
<b>ABSTRACT</b> . . . . .	xiv
<b>CHAPTER</b>	
<b>I. Some Philosophy</b> . . . . .	1
<b>II. Recovery of Robot Behaviors with Data</b> . . . . .	17
2.1 Introduction . . . . .	17
2.2 Templates and Anchors . . . . .	20
2.3 A Motivating Simulation . . . . .	22
2.3.1 Problem Statement . . . . .	22
2.3.2 Optimization . . . . .	25
2.3.3 Simulation Results . . . . .	27
2.4 Theory . . . . .	29
2.4.1 Geometric Mechanics . . . . .	30
2.4.2 Something Obvious . . . . .	32
2.4.3 The Kinematic Connection Case . . . . .	34
2.4.4 Generalization Away From Mechanics . . . . .	43
2.4.5 Canonical Cost Functions . . . . .	48
2.5 Crawler . . . . .	49
2.6 Enepod . . . . .	59
2.6.1 Methods . . . . .	62
2.6.2 Results . . . . .	74
2.7 Relationship with Other Learning Perspectives . . . . .	74



2.7.1	Kinematic Synergies . . . . .	74
2.7.2	Scaffold . . . . .	75
2.7.3	Planning . . . . .	76
2.7.4	Relationship with Output Tracking . . . . .	77
2.8	Summary . . . . .	77
2.9	Future Work . . . . .	78
<b>III. Event Selected Control and Analysis . . . . .</b>		<b>95</b>
3.1	Introduction . . . . .	95
3.2	Event Selected Systems . . . . .	97
3.2.1	Event Selected Vector Fields . . . . .	99
3.3	First Order Properties of Event Selected Systems . . . . .	105
3.4	Summary of $EC^r$ Systems . . . . .	108
3.5	Numerical Estimation of the Bouligand Derivative . . . . .	109
3.5.1	Factorial Complexity of $EC^r$ flows . . . . .	109
3.5.2	Notational Preliminaries . . . . .	111
3.5.3	Impact time order induces a cone decomposition . . . . .	112
3.5.4	Definition of the sorting map . . . . .	113
3.5.5	The corner flow . . . . .	115
3.5.6	Differential Equality . . . . .	119
3.5.7	Leveraging differential equality . . . . .	121
3.5.8	Computing the B-derivative with prefix vectors . . . . .	122
3.5.9	Computing prefix vectors without impact times . . . . .	125
3.6	Computational Procedures . . . . .	130
3.6.1	Fast integration of the corner flow . . . . .	130
3.6.2	Computing the B-derivative of the flow . . . . .	131
3.6.3	A note on computational complexity . . . . .	132
3.6.4	Validation . . . . .	133
3.6.5	Piecewise Constant Systems . . . . .	133
3.6.6	Stable Second Order Oscillator . . . . .	134
3.6.7	Planar Rigid Body with Contacts . . . . .	135
3.7	Summary of Numerical Methods . . . . .	138
3.8	Piecewise-Constant Feedback . . . . .	141
3.8.1	Consequences for Implementation . . . . .	143
3.8.2	General Polygonal Curves . . . . .	145
3.9	ESS and RHex . . . . .	147
3.9.1	Piecewise Constant Vector Fields via Chambered First-Order Holds . . . . .	152
3.9.2	Software Architecture and Data Protocol . . . . .	154
3.9.3	Simulation . . . . .	161
3.9.4	Hardware . . . . .	163
3.9.5	Performance . . . . .	169
3.10	Future Work . . . . .	172

<b>IV. Deadbeat Stabilization . . . . .</b>	<b>185</b>
4.1 Deadbeat Hopping . . . . .	185
4.1.1 Periodic Hopping . . . . .	189
4.1.2 Encoding control as a pre-transition constraint manifold . . . . .	190
4.1.3 Hybrid control system definition . . . . .	190
4.1.4 Controller design . . . . .	191
4.2 SLIP with Horizontal Velocity . . . . .	203
4.2.1 Limitations . . . . .	209
4.2.2 Implication For Mechanical Design . . . . .	214
4.3 Future Work . . . . .	216
<b>V. Conclusion . . . . .</b>	<b>226</b>
<b>APPENDICES . . . . .</b>	<b>228</b>

## LIST OF FIGURES

### Figure

2.1	The CT-SLIP with indicating variables. For clarity, only one leg is shown. A periodic gait is an concatenation of a sequence trajectory segments from holonomic subsystems. Reproduced from Fig 2.2 of <a href="#">Siepel and Holmes (2007)</a> . . . . .	23
2.2	Traces of the CoM and legs. The CoM is black, one leg is red, and the other is blue. Ten initial conditions in the stability basin of the original periodic orbit were sampled. For the goal system, all were stable (top). When perturbed, 8/10 crashed (middle). The recovery method attempted to find a single $\lambda_1^*$ such that all were stable. It succeeded only for 8/10 initial conditions. . . . .	29
2.3	The grey members indicate components that belong to the robot. The points $l_1$ and $l_2$ are fixed foot locations. Each joint $\theta_j$ is a powered rotational joint. The points $h_1$ and $h_2$ are the attachment points at which the limbs attach to the body. The center point of the foot locations $q$ defines the value $r$ and angle $\alpha$ (in red), which are our choice of encoding template. . . . .	50
2.4	$r$ and $\alpha$ coordinates of the template along the $\theta_{org}$ curve, i.e. - the desired reference trajectory on the template. . . . .	55
2.5	The six joint angles of $\theta_{org}$ . . . . .	55
2.6	The CoM group velocity $\dot{g} = (v_x, v_y, v_\theta)$ over time. The “desired” curve is the undamaged motion that we are aiming to recover. The blue “old” curve is the group velocity achieved if no recover strategy is attempted post damaged. The “recovered” trace is the performance after recovery. . . . .	56
2.7	Recovered joint angle inputs. The $\theta_1$ (blue) trace is visibly seen to be constant, reflecting the jammed state. . . . .	59
2.8	The enepod robot. The wooden sticks mount retroreflective markers that are used by a motion capture system to capture configuration data in real time. The motion capture system is part of the hardware-in-the-loop optimization setup used to generate motion. . . . .	61
2.9	Leg groupings on the Enepod- the (BL,MR,FL) legs maintain phase, while the (BR,ML,FL) are collectively out of phase with the first tripod. . . . .	62

2.10	Our parametric scheme for representing input signals. The “sig” trace is a desired nonlinear function, while “interp” is our linear interpolation of it. Our parameterization adjusts the amplitude of the knots, but not their horizontal location. The sawtooth approximation of a sine wave is operationally equivalent on the robot. . . . .	65
2.11	The filtered marker traces in world coordinates for an execution. The robot starts stationary, walks for a time, and then stops. . . . .	69
2.12	The same traces transformed into body coordinates using the virtual rigid mass. In these coordinates, the bias due to translation in the world frame is removed. . . . .	69
2.13	The summed vertical deflection signals for tripod one and two. The dashed lines are over a single stride, while the solid lines are Fourier-series ( $n=5$ ) fits of the measured data. . . . .	70
2.14	A schematic of how the vertical deflection angle is determined from marker data. The distal markers are used as an angular gain to compensate for the resolution of the motion capture system. . . . .	71
2.15	Time series data - the (BR,FR,ML) tripod signal is colored orange, while the (BL,MR,FL) tripod signal is shown in blue. The jump near data point 600 is where the robot changed directions. . . . .	72
2.16	Phase plots of the vertical spring deflection. The blue is the original data plotted as a function of phase. Yellow is after running a 2-order 2-sample discrete low pass filter twice, yellow is a 2nd order polynomial fit, while green is a 10th order Fourier series. We see considerable symmetry between tripods. Note that the (BL, MR, FL) tripod has been phase-shifted by $\pi$ for clarity. . . . .	73
2.17	$fa^i(t)$ for each limb, organized by phase. Darker colors are earlier values of phase, while warm colors are later phase values. We observe that there is not an apparent consistency between various limbs. . . . .	80
2.18	The singular values of the $fa^i$ data. It appears the data is predominately two-dimensional, but in such a way that does not correspond direct deflection of each tripod. . . . .	81
2.19	The first two principal components of the fore-aft deflection data. The dashed lines indicate measurement data, while the solid line is the Fourier-series approximation of the data. . . . .	82
2.20	Sagittal and aerial view of one leg. The red and yellow groupings of markers are used to determine a relative deflection . . . . .	83
2.21	A comparison of pre and post optimization. We the represented $\xi = (\xi_x, \xi_y, \xi_\theta)$ by the 2D position with $x = (\xi_x^2 + \xi_y^2)^{1/2}$ and $y = \xi_\theta \in [0, 2\pi]$ . We plotted the results of individual strides (the red pluses are the first ten percent of strides; the blue “x”’s are the last ten percent of strides; as well as the contours of a kernel smoothed density produced from those data (blue to cyan last points, black to orange the first ten percent of points . . . . .	84
2.22	The cost function along for the enepod after $N = 36$ iterations. . . . .	85

3.1	The $PC^r$ impact time map $\tau$ rectifies a general set of points that share the same transition sequence into a cone in $\tau$ coordinates. It additionally maps the vector field to a flowbox. The light yellow indicates a subset of points that all have transition sequence $M_1, M_2$ , while the green domain are points with $M_2, M_1$ as its transition sequence. The light-grey lines are solution curves, while the dark black lines are transition manifolds. . . . .	119
3.2	Each cone $\mathcal{K}_\sigma$ is spanned by a set of specially chosen prefix vectors. When mapped forward by the unit time flow, they create a pair of simplices. These simplices define a unique linear map that is the $D_x\tilde{\phi}(0, 0; u, v)$ for $\tau(u), \tau(v) \in \mathcal{K}_\sigma$ . . . . .	128
3.3	A quiver plot of the $F$ in the plane for $\nu = 1$ , $\delta = 0.6$ . In this case the original and corner flows coincide. The dashed line is the segment of a solution - it's transition sequence determines the value of $D\tilde{\phi}(0, 0; u, v)$ on the drawn simplices. The lower simplex is mapped to the upper one. Vertices with markers of shared type are paired by $\tilde{\phi}(1, \cdot)$ , and thus, by $D\tilde{\phi}$ . . . . .	134
3.4	Heat maps comparing the results from computing the B-derivative with our algorithm, versus numerically evaluating the known exact result. . . . .	135
3.5	A comparison of the numerical Jacobian and B-derivative representation for the initial condition $(-0. , 2.57, -0.001. , -0. , -8.2 , -0.)$ . . . . .	139
3.6	The chair model we simulated. The spring-damper is a restoring force that was generated based on the contact point of the chair. In the air, the spring-damper is undefined. Once a leg touched down, the spring-damper instantaneously appeared. . . . .	140
3.7	An illustration of locally stabilizing the diagonal. . . . .	143
3.8	Monotonic polygonal curve in the plane stabilized with canonical devices through a rectifying change of coordinates. . . . .	147
3.9	Monotonic polygonal curve in 3-space stabilized with canonical devices through a rectifying change of coordinates. . . . .	148
3.10	An image of our Rhex. . . . .	149
3.11	An illustration of a Buehler clock . . . . .	150
3.12	A two-dimensional ESS vector field on the torus. The off-diagonal dots are a collection of initial conditions that are mapped forward in time. The horizontal and vertical lines illustrate the event surfaces. Note that the green guard surfaces only change that coordinates' velocity component (e.g. horizontal green lines only change the horizontal component). These events, being only a function of one coordinate of the state, do not require global information. The other color of guards can change both components of the vector field. These indicate were a global event needs to be reported. . . . .	151
3.13	The periodic attractor plotted as a function of time. It features the analogous "fast-slow" behavior as a Buehler-clock. The horizontal lines indicate where the product guards (P-type) occur. . . . .	152

3.14	An illustration of the network topology and clock rates associated to RHex. M1,M2, and M3 are motor controllers, while “py” denotes the center CPU that interface with every controller. “10 Hz” is used to indicate low communication rates; since data is only irregularly sent per revolution, there is not an obvious way to make sense of the term “bandwidth”. . . . .	153
3.15	UML sequence diagram depicting the message flow sequencing as a function of time between the host CPU and a single motor. The commBuff classes are the linear buffers from above. SM is a state-machine that parses incoming data payloads into executable statements.	156
3.16	An example of Eqn. (3.32) as a Boolean decision. $\theta_1 = \pi - 0.1$ , $\theta_2 = \pi + 0.1$ , the window of which is indicated by the red vertical angles. The orange curve is the phase curve $\theta(t) = t$ , while the blue curve is the Guard value - when $\theta \in (\theta_1, \theta_2)$ , the function returns true.	158
3.17	A simulation output in the time domain of the virtual RHex. The blue and green triangular waveforms are the positions of each leg. The controller is grouping them well into two tripods. The lower square-like waves are the corresponding velocities. The jaggedness is a result of injected noise and model irregularities that are being rejected via the event-selected feedback. . . . .	162
3.18	A complete RHex leg. The fiber-glass leg is load-bearing, while it attaches to the BL-DC motor driver via a 3D printed fiber-reinforced nylon mount. The motor is bolted to the aluminum plate, which then attaches to the RHex frame. The visible wiring and electrical components are part of the optically coupled ground contact sensor. On the reverse, not pictured, is the motor controller and thermal safety circuitry. . . . .	164
3.19	An above view of the servo-module and RHex leg. (A) is the flexible LED strip wound around the motor mount, (B) is the thermal safety daughter board, while (C) is the ATC blade fuse. . . . .	165
3.20	A simple pulse generator – the push-button closure illuminates LED1 with a bandwidth controller via R1,R2, and C1. . . . .	166
3.21	An example trace of the resulting input signal to the microcontroller. The green trace is the signal produced by switch, while the yellow trace is the resulting output of the photo-edge detector. . . . .	167
3.22	The contact switch used to indicate if the leg has impacted the ground. When the beam is depressed by contact, the receiving circuitry converts this event into a 50 ms pulse. . . . .	168
3.23	The thermal safety schematic. There are three channels , one for each branch of the H-bridge, that are multiplexed together to TX line of the python kernel. Suppression of traffic on this line violates the heartbeat timer, resulting in a shutdown of the servomodule. .	169
3.24	Power module schematic. The function is each section is described in text. . . . .	170

3.25	Reference tracking an alternating tripod gait in dry-dock, where the legs do not make contact with the ground. The upper plot is the average angle of each tripod, while the lower figure is the velocity. . . . .	171
3.26	Reference Buehler tracking. The robot starts suspended in the air, and is then placed on the ground around data point 7000 to stride forwards. The high tracking quality degrades, as the legs in contact with the ground appear to experience a pure acceleration until their eventual liftoff. . . . .	172
3.27	Reference Buehler tracking. The blue trace is one marker, while the orange trace is the other. The upper plot is the $x$ coordinate of the plane, while the lower plot is the $y$ coordinate of the plane. Since the robot is not airborne, the $z$ coordinate is omitted. . . . .	173
3.28	Simulated trajectory data down-sampled via the host controller using P-type guards. Zero noise was added to the response. Finite-response time was enabled, smoothing the knot points. . . . .	174
3.29	The physical robot executing the same command sequence – while irregularities emerge, it is clear the same qualitative structure as the simulated response is present. . . . .	175
3.30	A mixed regime controller. The in-tripod synchronization is accomplished with distributed event-selected control, while the phase difference between the tripods is done with feedback. Red traces are one tripod, while blue traces are the other. We can see that a disruption to limbs is effectively rejected completely for in two strides. This data was collected in dry-dock. . . . .	175
4.1	A depiction of a SLIP model in gravity subject to uncertain contact. While a window of possible contacts is known, the precise point of contact is not. The control method proposed here presents a solution to preserve the gait of the SLIP. . . . .	186
4.2	A visual depiction of arrangement of hybrid domains. The submanifold $L$ (the zero level set of the goal surface, pulled back via the flow in $B$ ) is defined by the flow of the vector field defined in domain $B$ . If the dynamics of the $A$ system naturally have dynamics invariant to this surface, it will reject any uncertainty in switching manifold. . .	194
4.3	Vertical Hopper bouncing over a moving plate (height plotted against time), with an actuator (active only in flight) changing the rest length of the leg (pale red) and a muscle-like element (green) supporting the payload mass (dark grey). While the foot (light grey) is in contact with the ground, ground height remains constant; it changes instantaneously when the hopper is at its apex height. . . . .	195
4.4	The pullback of $g$ into $g^*$ into the region $X^*$ around $y_* = 1.571$ . The zero contour is the control objective in this region. Values were sampled on a grid with step size 0.012 . . . . .	198

4.5	Hopper simulation with our controller and randomly changing ground height. Each hop begins with a descent (red) associated with a changing “virtual” foot position (purple), ending when the foot intercepts the ground, triggering a transition into stance (blue), which persists until toe-off. After toe-off the hopper ascends (green) to a new apex; upon reaching it the ground height (black) may change. To demonstrate the deadbeat nature of this controller, the desired apex height for the next hop is indicated (light blue), and each ground height persists for two hops to show that hopping height reaches a new steady state after a single hop. . . . .	200
4.6	Phase-space plot of the same simulation as Fig. 4.5 . . . . .	201
4.7	The relative position of the internal state and the pre-image of 0 under $g_*$ . The difference between the two illustrates the necessary leg length to shift the descent trajectory pointwise to the contour. . . . .	202
4.8	A depiction of an execution of a Spring Load Inverted Pendulum (SLIP) with horizontal velocity. The relative displacement above the ground, and horizontal velocity at apex are being preserved. The spring dynamics are non-conservative, so energy can be added or removed. . . . .	204
4.9	The feedforward control law during descent for varying the leg angle and damping as a function of the vertical velocity state, it is implicitly a function of time. The leg angle is consistent with <a href="#">Seyfarth et al. (2006)</a> , in that a <i>retraction</i> improves the stability properties. . . .	208
4.10	A seven-stride sample gait using the feedforward controller. Note that the axes are not equal, due to the large horizontal distance covered per-step compared to the vertical compression. Here, the pink line illustrates the current position of the leg during descent, while the cyan is the leg during stance. During descent, the leg angle and damping are governed by figure 4.9. $\delta L = [0. \quad , -0.0179, -0.0302, -0.0071, 0.0093, -0.0190]$ . . . . .	210
4.11	A seven stride gait under identical control as figure 4.10. $\delta L = [0. \quad , 0.0060, 0.0473, -0.0018, 0.0332, 0.0417]$ . The leg is displayed less frequently compared to figure 4.10 for clarity - the controllers are still given by polynomials. . . . .	211
4.12	A gait where the controller has failed to stabilize the SLIP model with respect to ground height, illustrating that producing meaningful controller numerically may be computationally expensive to gain rich enough data sets for produce very reliable feedforward controllers. $\delta l = ([0. \quad , 0.0408, -0.0090, -0.0345, -0.0024, 0.0414])$ . .	212
4.13	Examples of transition sets. The left figure shows a sequence of level sets, each a manifold, so there is a clear way to pass a trajectory through them. The red arc denotes a solution of the underlying vector field. The figure on the right shows a transition surface that is not an embedded submanifold. Here, the point of common intersection would pose a problem. . . . .	214



## LIST OF TABLES

### Table

2.1	Parameters . . . . .	22
2.2	Parameters . . . . .	52
4.1	Parameters . . . . .	197
4.2	Parameters . . . . .	206

# LIST OF APPENDICES

## Appendix

A.	Extension of $EC^r$ fields . . . . .	229
B.	Asymptotic Phase for Limit Cycles . . . . .	232

## ABSTRACT

For robots to ever achieve significant autonomy, they need to be able to mitigate performance loss due to uncertainty, typically from a novel environment or morphological variation of their bodies. Legged robots, with their complex dynamics, are particularly challenging to control with principled theory. Hybrid events, uncertainty, and high dimension are all confounding factors for direct analysis of models. On the other hand, direct data-driven methods have proven to be equally difficult to employ. The high dimension and mechanical complexity of legged robots have proven challenging for hardware-in-the-loop strategies to exploit without significant effort by human operators. We advocate that we can exploit both perspectives by capitalizing on qualitative features of mathematical models applicable to legged robots, and use that knowledge to strongly inform data-driven methods. We show that the existence of these simple structures can greatly facilitate robust design of legged robots from a data-driven perspective. We begin by demonstrating that the factorial complexity of hybrid models can be elegantly resolved with computationally tractable algorithms, and establish that a novel form of distributed control is predicted. We then continue by demonstrating that a relaxed version of the famous templates and anchors hypothesis can be used to encode performance objectives in a highly redundant way, allowing robots that have suffered damage to autonomously compensate. We conclude with a deadbeat stabilization result that is quite general, and can be determined without

equations of motion.

## CHAPTER I

### Some Philosophy

Legged robots have received considerable attention from the robotics community for the past several decades. While advancements have been made in their design and control, a significant open challenge is incorporating robustness into their performance, especially with regards to environmental interaction. Despite the progress made, many legged robots are still prone to acute locomotive failure when exposed to a realistic environment. Moreover, almost no attention at all has been given to recovery strategies for robots that suffer damage - the dynamic models often employed in the design and control of robots are invariably assumed to be fixed during the operating lifetime of the device. When we review the copious examples of animals that employ legs as their primary means of motion and interaction, we see incredible dynamic resiliency and insensitivity to the environment and morphological variation due to injury or age, routinely outperforming conventional robotic platforms.

Roboticists typically employ dynamical systems and feedback control to generate and stabilize behaviors of interest out of robotic platforms. We observe that often the behaviors of interest, such a periodic gait, are intuitively *constrained* by the putative underlying dynamic model, rather than *defined* in such terms. That is, we often think of tasks as high level specifications such as “lift an object”, or “open a door”, rather than a trajectory of a differential equation as the design specification. Performance

goals for robots are often initially given as an relationship between inputs and outputs. In the case of using global, “exact”, dynamic models, wherein a robot is modeled as a single set of differential equations, the designers encode the behavioral specification by fiat into the equations of motion, often in the form of goal trajectories or attractive submanifolds.

Significant effort has been expended on the analysis of these encoded behaviors, but obstacles still present themselves. The mathematical theory of Lagrangian mechanics, which often model robots, (*Murray et al.*, 1994), (*Westervelt et al.*, 2003), (*Bloch et al.*, 2003) and control is mature. While powerful theoretical tools concerning stability, convergence, and tracking have been established, often actual hardware implementations resort to more simplistic such as reference tracking and PID control due to the difficulty implementing the more general results. The abstract theory’s utility as a tool to elicit a specific response from a legged robot is additionally confounded by the complexity of the dynamical equations used, or even the total absence of reliable models (e.g. - the difficulties in modeling contact dynamics of a limb striking the ground). Contacts present a particular challenge. If multiple contacts are to be expected (generically the case, if there are multiple limbs), the factorial number of sequences have not in-so-far had a compact description. Thus, mathematical robotics is forced to account for combinatorial complexity to analyze the stability and convergence of simultaneous contacts, and their dependency (or lack-there-of) on order. Generating control inputs that solve a high-order non-linear, non-smooth dynamical system through a factorial number of transitions becomes immensely challenging analytically, and designers often must employ numerical optimization, make intuitive leaps to synthesize controllers, and/or pre-define a restricted region of state space to occupy. (*Gupta and Pobil*, 1998), (*Sharir*, 1989), (*Ratliff et al.*, 2009), (*Koditschek and Rimon*, 1990), (*Westervelt et al.*, 2003) (*Da et al.*, 2016), (*Hereid et al.*, 2016), (*Ramezani et al.*, 2014)

Other researchers (*Tedrake et al.*, 2004) , (*Spröwitz et al.*, 2013), (*Reinkensmeyer et al.*, 2014),(*Pratihari et al.*, 2002) have successfully demonstrated high performance through the obviation of any approach based on model structure, instead relying on optimization methods treating the robot (either the equation of motions in simulation, or hardware-in-the-loop optimization) as a black-box that generates input-output sequences. Machine learning and optimization have yielded impressive demonstrations, but much remains to be done with regards to formal guarantees such as stability. Solving optimization problems in a realistic time frame is also still a key concern - while machine learning has been very fruitful on problems such as image recognition and classification, the complexity and constraints of a robot’s underlying dynamics (e.g., integration of the dynamics, either in simulation or via trial executions of a real platform) require non-trivial computational or experimental resources on problems of realistic complexity (*Mordatch et al.*, 2012b; *Bellman*, 1961). The high-dimension of full-state models leave methods that require a large number of iterations or cost function evaluations infeasible (*Kober et al.*, 2013; *Levine et al.*, 2018). Hardware-based optimization faces its own unique limitations in objective function evaluation - in the absence of a model, evaluating the cost function requires moving a physical device through space. The operating time and mechanical wear limit the number of evaluations that can be performed compared to a pure software routine (*Deisenroth et al.*, 2013; *Yosinski et al.*, 2011). Finally, while model-free optimization tools are powerful, their solutions provide little insight or understanding of any putative fundamental principles (*Tedrake et al.*, 2004). Akin to having an *answer* to a question rather than a *solution technique*, specific instances of trajectories on a given robot are difficult to abstract as principals that translate to another device.

Of substantial import, while we do *not* claim that the above approaches are *wrong*, we *do* assert that these approaches are often *hard*, and often rely as much on human cleverness as formal construction. Our main argument will be that by exploiting

simple structures associated with mechanical systems, stable dynamic locomotion can be achieved without the need to explicitly address high-dimension Lagrangian systems or perform extensive optimization.

We present three examples of exploiting or designing structural properties of mechanical systems to achieve improved robustness against uncertain environmental contacts or morphological variation. Each proposed method is validated computationally, illustrating that they are *effectively computable* strategies that are reasonably generic, yet still have formal guarantees. We eliminate the need to have detailed knowledge of the “true” underlying dynamical model of the robot, and merely assume its existence and form; We make use of qualitative properties, or observing functions that only provide input-output data.

The first section is focused on developing a method to preserve robot behaviors when the governing dynamics change through damage. We are interested in minimizing the recovery time required for a robot that undergoes irrevocable morphological damage, such as a limb breaking, or a motor seizing. By recording the observation variables of the undamaged system, and reconstituting them through control as a dynamical invariant on the damaged robot, we specify desired behaviors as high-codimension objects that admit an entire manifold of acceptable inputs. The functions define differential constraints on a virtual manifold that completely defines a target robot motion. By lifting these constraints into the robot, a full description of acceptable inputs that achieve the goal is defined using less variables than originally required, reducing the complexity of the recovery task.

As such, rather than attempting to preserve a specific trajectory (as such a thing may be undefined or impossible due to the altered dynamics), we are using control to induce a nonholonomic affine constraint on configuration space, requiring that the controlled vector field decompose into a mandated component, and a free component. The unconstrained dimensions admit a continuum of solutions compared to a



full trajectory constraint. The lifted constraints are additionally virtual constraints – the degree of fit is tunable by the user. While meeting this constraints precisely regenerates a desired behavior, relaxing this requirement allows approximate reconstruction. Much akin to how limping is achieved almost instantaneously in legged organisms that suffer mild injury, we can achieve “good enough” recovery by relaxing the constraint tolerance to gracefully specify the quality of recovery.

By conducting the above recovery procedure on a multi-limbed dynamic robot, we hope to demonstrate that a human-centric definition of “acceptable performance” is looser than a full trajectory constraint, and by exploiting this freedom, the difficulty of the recovery strategy is much reduced. We test this notion in a simulation of a two-legged clock-torqued spring-loaded inverted pendulum to regenerate a periodic gait, and on a multi-legged dynamic robot. We have constructed a six legged modular robot that has legs constructed out of spring steel, allowing a measurable energy exchange between the limbs and center of mass when the robot is executing a periodic gait. By synthesizing a reduction map on the robot into a four dimensional space with virtual constraints, we demonstrate that a desired gait can be found via optimization extremely rapidly - only 36 iterations. Intuitively, it might seem unreasonable that more constraints simplify a problem – after all, rubbing ones head and patting ones stomach seem much more challenging either tasks alone, but we instead develop constraints that encode helpful, rather than adversarial, information of how inputs and outputs are related. E.g., unlike the above demonstration of agility, our constraints are more akin to being told that to ride a bike forward, one should restrict body motions that keep the bike upright. While it’s certainly *possible* to engage in theatrical maneuvers with the bicycle, we show that we can encode behaviors that have useful low-dimensional components.

The second, based on a recently defined class of hybrid dynamical systems called “event-selected systems” explicitly incorporates contacts into the equations of motion.

We will exploit that a piecewise-constant vector field analogous to the differential linearization of a smooth system can be derived using an extended notion of calculus, with associated numerical routines to estimate a piecewise affine operator. We will also highlight results that demonstrate this piecewise linear operator has analogous features to smooth linearization for characterization of local dynamic properties. We will construct a numerical algorithm that eliminates the apparent factorial complexity with contact sequencing, illustrating that the category of event-selected models have substantial practical mileage over other approaches to hybrid dynamics.

We demonstrate that we can employ contacts to synthesize gaits using a low-complexity distributed controller that is distinct from traditional reference trajectory tracking, greatly simplifying the communication complexity associated to tracking a desired trajectory. Despite the usage of only piecewise-constant vector fields and distributed control with little communication, event selected vector field theory provides stability and robustness guarantees.

The final section demonstrates that certain classes of periodic hybrid dynamical systems undergoing uncertain transitions generically admit feedforward deadbeat controllers ([Council et al., 2014](#)). The feedforward controllers stabilize a system undergoing irrevocable, but poorly predicted, contact. That is the contact is assumed to occur from physical constraints (gravity, etc), but the exact time of the contact is allowed to vary between executions. By computing a feedforward policy that is initialized at the start of a given execution, the terminal state (e.g., a transverse section in the final domain) is made invariant to the variable contact. While other groups ([Hutter et al., 2010](#)), ([Yu et al., 2012](#)) have discovered this phenomena for specific devices, namely hopping robots and juggling robots ([Reist and D'Andrea, 2012](#)), as far as we are aware, each used model-specific knowledge in their analysis. We show that the existence of deadbeat controllers in this sense is reasonably general for a large class of hybrid systems. We also articulate that the construction of them is easily derived

from the implicit function theorem. An immediate consequence of this approach is that, rather than use any model knowledge whatsoever to infer a deadbeat controller, we can instead experimentally derive the controller without ever explicitly attempting to express the system’s dynamics as a set of differential equations.

Together, we aim to convince the reader that space between very principled hybrid Lagrangian dynamics, and very blind black-box optimization, contains useful control and design principles motivated by the qualitative structure of the otherwise unknown or uncertain dynamics for legged robots. Furthermore, these principles can be realized into tools that are easier to implement than contemporary paradigms in robotic control. I.e., we can develop approaches to produce resilient legged robots using more structure than black-box optimization, but do not require full dynamic models.

## BIBLIOGRAPHY

### Bibliography

(2019), *ATOF Blade Fuses Rated 32*, LittleFuse.

Adolph, K. E., and S. R. Robinson (2013), The road to walking: What learning to walk tells us about development, *Oxford handbook of developmental psychology*, 1, 403–443.

Aizerman, M., and F. Gantmacher (1958), Determination of stability by linear approximation of a periodic solution of a system of differential equations with discontinuous right-hand sides, *The Quarterly Journal of Mechanics and Applied Mathematics*, 11(4), 385–398.

Alexander, K., et al. (1994), *Smooth invariant manifolds and normal forms*, vol. 7, World Scientific.

Alexander, R. M. (1984), The gaits of bipedal and quadrupedal animals, *The International Journal of Robotics Research*, 3(2), 49–59.

Ankarali, M., and U. Saranli (2010), Stride-tostride energy regulation for robust self-stability of torque-actuated dissipative spring-mass hopper, *Chaos*, 20(033121), doi:10.1063/1.3486803.

Arnold, V. (1973), *Ordinary differential equations*, The MIT Press, Cambridge, Massachusetts.

Arslan, O., and U. Saranli (2012), Reactive planning and control of planar spring-mass running on rough terrain, *IEEE Trans. Robotics*, 28, 567–579, doi:10.1109/TRO.2011.2178134.

Bellman, R. E. (1961), Dynamic programming treatment of the traveling salesman problem.

- Bernardo, M., C. Budd, A. R. Champneys, and P. Kowalczyk (2008), *Piecewise-smooth dynamical systems: theory and applications*, vol. 163, Springer Science & Business Media.
- Bizzarri, F., A. Brambilla, and G. S. Gajani (2013), Lyapunov exponents computation for hybrid neurons, *Journal of computational neuroscience*, *35*(2), 201–212.
- Blickhan, R. (1989a), The spring-mass model for running and hopping, *J. Biomechanics*, *22*(11-12), 1217–1227, doi:10.1016/0021-9290(89)90224-8.
- Blickhan, R. (1989b), The spring-mass model for running and hopping, *J. Biomechanics*, *22*(11-12), 1217–1227, doi:10.1016/0021-9290(89)90224-8.
- Bloch, A., J. Baillieul, P. Crouch, J. E. Marsden, D. Zenkov, P. S. Krishnaprasad, and R. M. Murray (2003), *Nonholonomic mechanics and control*, vol. 24, Springer.
- Bloch, A. M., P. Krishnaprasad, J. E. Marsden, and R. M. Murray (1996), Nonholonomic mechanical systems with symmetry, *Archive for Rational Mechanics and Analysis*, *136*(1), 21–99.
- Bogert, A., K. Gerritsen, and G. Cole (1998), Human musclemodelling from a user’s perspective, *J. Elecromyogr. Kines.*, *8*(2), 119–124, doi:10.1016/S1050-6411(97)00028-X.
- Bongard, J., V. Zykov, and H. Lipson (2006), Resilient machines through continuous self-modeling, *Science*, *314*(5802), doi:10.1126/science.1133687.
- Bongard, J. C. (2011), Morphological and environmental scaffolding synergize when evolving robot controllers: artificial life/robotics/evolvable hardware, in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pp. 179–186, ACM.
- Browder, F. E. (1954), Covering spaces, fibre spaces, and local homeomorphisms, *Duke Math. J.*, *21*(2), 329–336, doi:10.1215/S0012-7094-54-02132-8.
- Bullo, F., A. D. Lewis, and K. M. Lynch (2002), Controllable kinematic reductions for mechanical systems: concepts, computational tools, and examples, in *Mathematical Theory of Networks and Systems*, vol. 124.
- Burden, S. A., S. Revzen, and S. S. Sastry (2015), Model reduction near periodic orbits of hybrid dynamical systems, *IEEE Transactions on Automatic Control*, *60*(10), 2626–2639.
- Burden, S. A., S. S. Sastry, D. E. Koditschek, and S. Revzen (2016), Event-selected vector field discontinuities yield piecewise-differentiable flows, *SIAM Journal on Applied Dynamical Systems*, *15*(2), 1227–1267.
- Carver, S., N. Cowan, and J. Guckenheimer (2009), Lateral stability of the spring-mass model suggests a two-step control strategy for running, *CHAOS*, *19*, doi:10.1063/1.3127577.

- Council, G., S. Yang, and S. Revzen (2014), Deadbeat control with (almost) no sensing in a hybrid model of legged locomotion, in *Advanced Mechatronic Systems (ICAMEchS), 2014 International Conference on*, pp. 475–480, IEEE.
- Cully, A., J. Clune, D. Tarapore, and J. Mouret (2014), Robots that can adapt like animals, *Nature*, *521*(7553), doi:10.1038/nature14422.
- Da, X., O. Harib, R. Hartley, B. Griffin, and J. W. Grizzle (2016), From 2d design of underactuated bipedal gaits to 3d implementation: Walking with speed tracking, *IEEE Access*, *4*, 3469–3478.
- Deisenroth, M. P., G. Neumann, J. Peters, et al. (2013), A survey on policy search for robotics, *Foundations and Trends® in Robotics*, *2*(1–2), 1–142.
- Dieci, L., and L. Lopez (2011), Fundamental matrix solutions of piecewise smooth differential systems, *Mathematics and Computers in Simulation*, *81*(5), 932–953.
- Ernst, M., H. Geyer, and R. Blickhan (2011), Spring-legged locomotion on uneven ground: A control approach to keep the running speed constant, in *Int. Conf. on Climbing and Walking Robots*, pp. 639–644.
- Farley, C., R. Blickhan, and C. Taylor (1985), Mechanics of human hopping : model and experiments, *Am. Zool.*, *25*, 54A.
- Fenichel, N. (1974), Asymptotic stability with rate conditions for dynamical systems, *Bulletin of the American Mathematical Society*, *80*(2), 346–349.
- Fenichel, N. (1977), Asymptotic stability with rate conditions, ii, *Indiana University Mathematics Journal*, *26*(1), 81–93.
- Filippov, A. F. (2013), *Differential equations with discontinuous righthand sides: control systems*, vol. 18, Springer Science & Business Media.
- Full, R. J., and D. E. Koditschek (1999), Templates and anchors: neuromechanical hypotheses of legged locomotion on land, *Journal of experimental biology*, *202*(23), 3325–3332.
- Galewski, M., and M. Koniorczyk (2016), *Global invertibility theorems and their applications-a variational approach*, Wydawnictwo Politechnika Łódzka.
- Galloway, K. C., G. C. Haynes, B. D. Ilhan, A. M. Johnson, R. Knopf, G. Lynch, B. Plotnick, M. White, and D. E. Koditschek (2010), X-rhex: A highly mobile hexapedal robot for sensorimotor tasks, *Tech. rep.*, University of Pennsylvania.
- Ghigliazza, R., R. Altendorfer, P. Holmes, and D. Koditschek (2004), A simply stabilized running model, *SIAM J. App. Dyn. Systems*, *2*(2), 187–218, doi:10.1137/S1111111102408311.

- Ghigliazza, R. M., and P. Holmes (2005), Towards a neuromechanical model for insect locomotion: Hybrid dynamical systems, *Regular & Chaotic Dynamics*, 10(2), 193–225, doi:10.1070/RD2005v010n02ABEH000311.
- Goebel, R., R. Sanfelice, and A. Teel (2009), Hybrid dynamical systems, *Control Systems, IEEE*, 29(2), 28–93, doi:10.1109/MCS.2008.931718.
- Golubitsky, M., and V. Guillemin (2012), *Stable mappings and their singularities*, vol. 14, Springer Science & Business Media.
- Groff, R. E., P. P. Khargonekar, and D. E. Koditschek (2003), A local convergence proof for the minvar algorithm for computing continuous piecewise linear approximations, *SIAM journal on numerical analysis*, 41(3), 983–1007.
- Guckenheimer, J. (), Isochrons and phaseless sets, *Journal of Mathematical Biology*, 1(3), 259–273, doi:10.1007/BF01273747.
- Guckenheimer, J., and S. Johnson (1995), *Hybrid systems II*, 202–225 pp., Springer-Verlag, London, UK.
- Gupta, K., and A. P. Pobil (1998), *Practical motion planning in robotics: Current approaches and future directions*, John Wiley & Sons, Inc.
- Hairer, E., G. Wanner, and P. Norsett (1993), Solving ordinary differential equations i – nonstiff problems, Springer Series in Computational Mathematics, 2 ed., Springer-Verlag.
- Hatton, R. L., and H. Choset (2011), An introduction to geometric mechanics and differential geometry.
- Hatton, R. L., and H. Choset (2013), Geometric swimming at low and high reynolds numbers, *IEEE Transactions on Robotics*, 29(3), 615–624.
- Hereid, A., E. A. Cousineau, C. M. Hubicki, and A. D. Ames (2016), 3d dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics, in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pp. 1447–1454, IEEE.
- Hill, A. (1938), The heat of shortening and the dynamic constants of muscle, *Proc. R. Soc. Lond. B*, 126(843), 136–195, doi:10.1098/rspb.1938.0050.
- Hirsch, M. W., R. L. Devaney, and S. Smale (1974), *Differential equations, dynamical systems, and linear algebra*, vol. 60, Academic press.
- Hirsch, M. W., C. C. Pugh, and M. Shub (2006), *Invariant manifolds*, vol. 583, Springer.
- Hirsch, M. W., S. Smale, and R. L. Devaney (2012), *Differential equations, dynamical systems, and an introduction to chaos*, Academic press.

- Holmes, P., R. Full, D. Koditschek, and J. Gukenheimer (2006a), The dynamics of legged locomotion : models, analyses, and challenges, *SIAM Review*, 48(2), 206–304, doi:10.1137/S0036144504445133.
- Holmes, P., R. Full, D. Koditschek, and J. Gukenheimer (2006b), The dynamics of legged locomotion : models, analyses, and challenges, *SIAM Review*, 48(2), 206–304, doi:10.1137/S0036144504445133.
- Hutter, M., R. C.D. M. Hoepflinger, and R. Seigwart (2010), Full state control of a slip model by touchdown detection, in *13th International Conference on Climbing and Walking Robots*, pp. 533–540.
- Ivanov, A. (1998), The stability of periodic solutions of discontinuous systems that intersect several surfaces of discontinuity, *Journal of Applied Mathematics and Mechanics*, 62(5), 677–685.
- Jarque-Bou, N., V. Gracia-Ibáñez, J.-L. Sancho-Bru, M. Vergara, A. Pérez-González, and F. Andrés (2016), Using kinematic reduction for studying grasping postures. an application to power and precision grasp of cylinders, *Applied ergonomics*, 56, 52–61.
- Jarque-Bou, N. J., A. Scano, M. Atzori, and H. Müller (2019), Kinematic synergies of hand grasps: a comprehensive study on a large publicly available dataset, *Journal of neuroengineering and rehabilitation*, 16(1), 63.
- Kelly, S. D., and R. M. Murray (1995), Geometric phases and robotic locomotion, *Journal of Robotic Systems*, 12(6), 417–431.
- Kenneally, G., A. De, and D. E. Koditschek (2016), Design principles for a family of direct-drive legged robots, *IEEE Robotics and Automation Letters*, 1(2), 900–907.
- Khalil, H. (2002a), *Non-Linear System: Third Edition*, Prentice Hall, Upper Saddle River, New Jersey.
- Khalil, H. (2002b), *Nonlinear systems*, 3 ed., Prentice Hall, Upper Saddle River, New Jersey.
- Kobayashi, S., and K. Nomizu (1963), *Foundations of differential geometry*, vol. 1, New York.
- Kober, J., J. A. Bagnell, and J. Peters (2013), Reinforcement learning in robotics: A survey, *The International Journal of Robotics Research*, 32(11), 1238–1274.
- Koditschek, D. E., and E. Rimon (1990), Robot navigation functions on manifolds with boundary, *Advances in applied mathematics*, 11(4), 412–442.
- Koon, W. S., and J. E. Marsden (1997), The geometric structure of nonholonomic mechanics, in *Proceedings of the 36th IEEE Conference on Decision and Control*, vol. 5, pp. 4856–4861, IEEE.



- Kuhn, H. W. (1960), Some combinatorial lemmas in topology, *IBM Journal of research and development*, 4(5), 518–524.
- Lee, J. M. (2013), Smooth manifolds, in *Introduction to Smooth Manifolds*, pp. 1–31, Springer.
- Levine, S., P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen (2018), Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection, *The International Journal of Robotics Research*, 37(4-5), 421–436.
- Lygeros, J., K. H. Johansson, S. N. Simic, J. Zhang, and S. S. Sastry (2003), Dynamical properties of hybrid automata, *IEEE Transactions on automatic control*, 48(1), 2–17.
- Marsden, J. E., R. Montgomery, and T. S. Ratiu (1990), *Reduction, symmetry, and phases in mechanics*, vol. 436, American Mathematical Soc.
- Meigniez, G. (2002), Submersions, fibrations and bundles, *Transactions of the American Mathematical Society*, 354(9), 3771–3787.
- Mordatch, I., Z. Popović, and E. Todorov (2012a), Contact-invariant optimization for hand manipulation, in *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*, pp. 137–144, Eurographics Association.
- Mordatch, I., E. Todorov, and Z. Popović (2012b), Discovery of complex behaviors through contact-invariant optimization, *ACM Transactions on Graphics (TOG)*, 31(4), 43.
- Müller, P. C. (1995), Calculation of lyapunov exponents for dynamic systems with discontinuities, *Chaos, Solitons & Fractals*, 5(9), 1671–1681.
- Murray, R., S. Sastry, and Z. Li (1994), *A mathematical introduction to robotic manipulation*, 1 ed., CRC Press.
- Murray, R. M. (2017), *A mathematical introduction to robotic manipulation*, CRC press.
- OLECH, C. (1998), On the wazewski equation, *ZESZYTY NAUKOWE-UNIwersytetu Jagiellońskiego-ALL SERIES-*, 1223, 55–64.
- Orhon, H. E. (2018), Model-based identification and control of a one-legged hopping robot, *arXiv preprint arXiv:1802.09634*.
- Ostrowski, J., and J. Burdick (1998), The geometric mechanics of undulatory robotic locomotion, *The international journal of robotics research*, 17(7), 683–701.
- Ostrowski, J. P. (1996), The mechanics and control of undulatory robotic locomotion, Ph.D. thesis, California Institute of Technology.

- Palmer III, L. R., and C. Eaton (2014), Periodic spring-mass running over uneven terrain using feedforward control of landing conditions, *Bioinspiration and Biomimetics*, 9(3), doi:10.1088/1748-3182/9/036018.
- Pepy, R., A. Lambert, and H. Mounier (2006), Path planning using a dynamic vehicle model, in *2006 2nd International Conference on Information & Communication Technologies*, vol. 1, pp. 781–786, IEEE.
- Pepyne, D. L., and C. G. Cassandras (2000), Optimal control of hybrid systems in manufacturing, *Proceedings of the IEEE*, 88(7), 1108–1123.
- Pratihari, D. K., K. Deb, and A. Ghosh (2002), Optimal path and gait generations simultaneously of a six-legged robot using a ga-fuzzy approach, *Robotics and Autonomous Systems*, 41(1), 1–20.
- Radford, J., and J. Burdick (1998), Local motion planning for nonholonomic control systems evolving on principal bundles, *A, A, 1, 3*.
- Ramezani, A., J. W. Hurst, K. A. Hamed, and J. W. Grizzle (2014), Performance analysis and feedback control of atrias, a three-dimensional bipedal robot, *Journal of Dynamic Systems, Measurement, and Control*, 136(2), 021,012.
- Ratliff, N., M. Zucker, J. A. Bagnell, and S. Srinivasa (2009), Chomp: Gradient optimization techniques for efficient motion planning, in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pp. 489–494, IEEE.
- Reinkensmeyer, D. J., et al. (2014), Tools for understanding and optimizing robotic gait training.
- Reist, P., and R. D’Andrea (2012), Design and analysis of a blind juggling robot, *IEEE Transactions on Robotics*, 28(6), 1228–1243.
- Revzen, S., and J. Guckenheimer (2008), Estimating phase of synchronized oscillators, *Phys. Rev. E*, 78(5), 051,907, doi:10.1103/PhysRevE.78.051907.
- Revzen, S., and M. Kvalheim (2015), Data driven models of legged locomotion, in *SPIE Defense+ Security*, pp. 94,671V–94,671V, International Society for Optics and Photonics.
- Revzen, S., D. E. Koditschek, and R. J. Full (2009), Towards testable neuromechanical control architectures for running, in *Progress in Motor Control*, pp. 25–55, Springer.
- Revzen, S., M. Kvalheim, S. Wilshon, and J. Guckenheimer (2018), Estimating phase from observed trajectories, *In preparation*, pp. 1–20.
- Rockafellar, R. T. (2003), A property of piecewise smooth functions, *Computational Optimization and Applications*, 25(1-3), 247–250.
- Ruff, C. (2002), Variation in human body size and shape, *Annual Review of Anthropology*, 31, 211–232, doi:10.1146/annurev.anthro.31.040402.085407.

- Rumbaugh, J., I. Jacobson, and G. Booch (2004), *Unified modeling language reference manual, the*, Pearson Higher Education.
- Saranli, U., M. Buehler, and D. E. Koditschek (2001), Rhex: A simple and highly mobile hexapod robot, *The International Journal of Robotics Research*, 20(7), 616–631.
- Sastry, S. (1999), *Nonlinear systems : analysis, stability, and control*, Springer.
- Schmitt, J., and P. Holmes (2000), Mechanical models for insect locomotion: dynamics and stability in the horizontal plane i. theory, *Biological cybernetics*, 83(6), 501–515.
- Scholtes, S. (2012), *Introduction to piecewise differentiable equations*, Springer Science & Business Media.
- Seyfarth, A., H. Geyer, M. Gunther, and R. Blickhan (2002), A movement criterion for running, *J. Biomechanics*, 35, 649–655, doi:10.1016/S0021-9290(01)00245-7.
- Seyfarth, A., H. Geyer, and H. Herr (2006), Swing-leg retraction : a simple control model for stable running, *J. of Exp. Bio.*, 206, 2547–2555, doi:10.1242/jeb.00463.
- Sharir, M. (1989), Algorithmic motion planning in robotics, *Computer*, 22(3), 9–19.
- Siepel, J., and P. Holmes (2007), A simple model for clock-actuated legged locomotion, *Regular and Chaotic Dynamics*, 12(5), 502–520, doi:10.1134/S1560354707050048.
- Simić, S. N., K. H. Johansson, S. Sastry, and J. Lygeros (2000), Towards a geometric theory of hybrid systems, in *International Workshop on Hybrid Systems: Computation and Control*, pp. 421–436, Springer.
- Spivak, M. (1965), *Calculus on Manifolds*, Perseus Books Publishing, L. L. C.
- Spröwitz, A., A. Tuleu, M. Vespignani, M. Ajallooeian, E. Badri, and A. J. Ijspeert (2013), Towards dynamic trot gait locomotion: Design, control, and experiments with cheetah-cub, a compliant quadruped robot, *The International Journal of Robotics Research*, 32(8), 932–950.
- Tassa, Y., T. Erez, and E. Todorov (2012), Synthesis and stabilization of complex behaviors through online trajectory optimization, in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4906–4913, IEEE.
- Tedrake, R., T. W. Zhang, and H. S. Seung (2004), Stochastic policy gradient reinforcement learning on a simple 3d biped, in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3, pp. 2849–2854, IEEE.
- Utkin, V. (1977), Variable structure systems with sliding modes, *IEEE Transactions on Automatic control*, 22(2), 212–222.

- Vejdani, H., Y. Blum, M. Daley, and J. Hurst (2013), Bio-inspired swing leg control for spring-mass robots running on ground with unexpected height disturbance, *Bioinspiration and Biomimetics*, 8(4), doi:10.1088/1748-3182/8/4/046006.
- Westervelt, E. R., J. W. Grizzle, and D. E. Koditschek (2003), Hybrid zero dynamics of planar biped walkers, *IEEE transactions on automatic control*, 48(1), 42–56.
- Winters, J. (1990), Hill-based muscle models: a systems engineering perspective, in *Multiple Muscle Systems*, edited by J. Winters and S.-Y. Woo, pp. 69–93, Springer New York, doi:10.1007/978-1-4613-9030-5\_5.
- Yosinski, J., J. Clune, D. Hidalgo, S. Nguyen, J. C. Zagal, and H. Lipson (2011), Evolving robot gaits in hardware: the hyperneat generative encoding vs. parameter optimization., in *ECAL*, pp. 890–897.
- Yu, H., M. Li, W. Guo, and H. Cai (2012), Stance control of the slip hopper with adjustable stiffness of leg spring, in *Mechatronics and Automation (ICMA), 2012 International Conference on*, pp. 2007–2012, IEEE.

## CHAPTER II

# Recovery of Robot Behaviors with Data

### 2.1 Introduction

The preservation of motion against structural changes is an ability many legged animals display with great aptitude - through injury, age, or the otherwise mutable nature of organic tissue, animals are able to recover unless the perturbation is, tautologically, “too severe”.

Robots would ostensibly be advantaged if they had similar ability to preserve task execution through damage or changes. Specifically, if an autonomous, or nearly-autonomous algorithm can be designed so that a robot is able to procedurally mitigate system failures inflicted by damage . Even though a morphological or structural change to a robot may alter the dynamics of the device, if the robot has enough redundant degrees of freedom, we would expect task preservation to be possible. Robots recovering from structural uncertainty has been explored before, at least in the world of locomotion. Cully et. al used an intelligent trial-and-error method on a walking robot to explore a pre-computed set of acceptable behaviors to produce a recovery behavior (*Cully et al.*, 2014). Bognard, Zykov, and Libson developed a walking robot that recovered from damage through continuous self-modeling to compensate and generate a new gait. (*Bongard et al.*, 2006).

We distinguish our approach by taking advantage of two dimension-reducing con-

structions. The first is the *templates and anchors* hypothesis (see [Full and Koditschek \(1999\)](#) for a detailed introduction). Briefly, it asserts that animals with complex models (the *anchor*) have coupled or dependent state variables that behave “as if” the robot had a lower-dimensional model (the *template*). In other words, if  $f : M \rightarrow \mathbf{T}M$  is the anchor vector field, there exists an attracting invariant immersed submanifold  $N \subset M$  (the template),  $\dim(N) < \dim(M)$  such that  $f|_N : N \rightarrow \mathbf{T}N$ . Considering the anchor as an invariant submanifold has many desirable properties, but it is also a strong requirement. A simpler approach, but one with weaker mathematical guarantees, is to consider the template as a manifold  $N$  with vector field  $g : N \rightarrow \mathbf{T}N$ , with submersion  $\varphi : M \rightarrow N$  such that  $g = \varphi_* f$ . In this regard, we think of the template as a virtual system whose trajectories are the “shadows” of those of  $f$ .

In this language, the behavior of interest is represented by a solution  $\phi^t(r_0) \in N$ . For example, if a cockroach has a Spring-Loaded Inverted Pendulum (SLIP) ([Blickhan, 1989a](#)) as a template, the behavior of the SLIP is what we wish to preserve. In face of damage to the anchor, the robot may retain sufficient control authority to implement a SLIP.

The second perspective we take advantage of is offered by geometric mechanics, and nearly equivalently, affine control systems. For Lagrangian systems with non-holonomic constraints that are symmetric under a group action, locomotion can be represented by a *connection* on a *principal fiber bundle*. ([Ostrowski and Burdick, 1998](#); [Bloch et al., 2003, 1996](#)). The resulting *reconstruction equation* neatly expresses the motion in the group as a function of internal shape variables. The total displacement of a planar robot moving its limbs can be thought of as an element  $g \in SE(2)$  resulting from cyclic limb motions  $l(t), t \in [0, T]$ . The “shape” of the robot is modulated to effect displacement of the center-of-mass.

In the sequel we will review the details of these constructions more carefully. The essence of them is that for certain robots, while high-fidelity models may be complex,

we can project away much of the apparent complexity by restricting our interest to trajectories that satisfy a known collection of constraints.

Before proceeding further, we might ask if a recovery strategy for a robot to autonomously (or even non-autonomously) compensate for a damaged limb is necessary if a technician is standing-by to take the robot offline and repair the damage. It would be immediately clear to any critical observer that such a strategy is unrealistic for a device that is intended to have any degree of substantial autonomy (e.g., a Martian rover). If a robot is faced with the possibility of remote failure in a complex environment with unknown obstacles, robust recovery methods need to be generic, agnostic to a particular failure mode or environment. Additionally, an autonomous device may need to produce a strategy quickly – the range of solutions to a stalled car on the train tracks depends critically on whether there is impending doom or not.

Furthermore, organisms exhibit sensitivity to quality of recovery – if a leg muscle is damaged, a human will almost immediately produce a compensatory gait, such as limping, to minimize pain while granting some mobility. Imperfect recovery, but with some functionality, is preferred over waiting for the muscle to heal entirely before walking again. With the above perspective, it is natural to consider recovery as an optimization problem. An in-situ workable solution “closer” to a true optima may be an adequate behavior in the short term, and can serve as initial data for a more complex or lengthy optimization offline.

The chapter is laid out as follows. We will start in §2.2 by reviewing the templates and anchors hypothesis, as well defining asymptotic phase. Armed with phase, we will continue in §2.3 with a motivating simulation that introduces virtual constraints on a CT-SLIP model. We will show how to recover from parametric damage, but see that the approach takes advantage of model-specific features.

Sec. §2.4 will transplant the lessons of the CT-SLIP to principal fiber bundles, and develop the associated theoretical results. We will then expand our theory by

extending our method to define a “behavioral specification” that is much more general. We will show that it both naturally includes our previous cases, and is compatible with other perspectives from control and mechanics. In Sec §2.5, we will apply our theoretical results to a kinematic crawling robot in simulation, demonstrating perfect recovery by applying our technique directly. We will additionally see that a key feature of this construction is that can be achieved without model information. Then, in §2.6, we use this observation to demonstrate our approach on an multi-legged robot in hardware using limited model information. The constraints can be constructed using measurements of the functional system, and we use reinforcement learning to design controllers to meet these constraints through hardware-in-the-loop optimization. By using our method, we will show that a cost function defined via the constraints can move effectively toward a goal behavior using only a few iterations of a Nelder-Mead optimizer.

## 2.2 Templates and Anchors

If we abide by the model that a template is a normally attractive invariant manifold (NAIM), we can conclude the existence of asymptotic phase. Asymptotic phase is a concept that has received considerable attention in the dynamics community ([Fenichel, 1974](#); [Alexander et al., 1994](#); [Hirsch et al., 2006](#); [Fenichel, 1977](#)), and we briefly summarize its definition and some essential properties here.

Let  $M$  be a smooth manifold with vector field  $f : M \rightarrow \mathbf{TM}$  with flow  $\phi^t(\cdot)$ . Suppose that the manifold  $N \subset M$  is asymptotically stable under  $\phi^t(\cdot)$ . If there exists smooth submersion  $P : M \rightarrow N, P(M) = N$  such that  $\forall x \in M$ , and for any  $q \in N, q \neq P(x)$ ,

$$\lim_{t \rightarrow \infty} \frac{\|\phi(x, t) - \phi(P(x), t)\|}{\|\phi(x, t) - \phi(q, t)\|} = 0 \quad (2.1)$$

We call  $P$  the phase map, and the value  $P(x)$  the phase of  $x$ . The point  $P(x)$  is



unique in the sense that  $x$  converges with  $P(x)$  more rapidly than with any other putative point  $q$ .  $P$  is a nonlinear projection, as  $P \circ P = P$ . Points that are in-phase have the same asymptotic behavior. Stable normally hyperbolic manifolds always have asymptotic phase. Points that are in phase asymptotically approach each other as  $t \rightarrow \infty$ . Geometrically, the projected point is the “shadow” of infinity many points that share the same phase.

It is this last property that we wish to carry by analogy our simpler case of having the anchor  $M$  and the template  $N$  be separate manifolds. Asymptotic phase identifies anchor states with template states. If  $N$  is a NAIM, it also conveys dynamic information. However, if  $\varphi : M \rightarrow N$  is merely any submersion, it is a non-linear projection that allows us to *explicitly* relate the state of the template with a family of equivalent states in the anchor, regardless of how the dynamics of the two systems are related. In the sequel we will employ such phase maps to define *encoding templates*. Distinct from the regular template above, an encoding template does not have dynamics conjugate to the anchor. It will function as a collection of *observation variables* that will be shown to fully characterize a desired motion more parsimoniously than the state of the anchor.

In the sequel, we will reserve then notation  $\varphi$  for submersions in this context, and will use the terminology “phase map” for it, whether or not it is properly asymptotic phase. We will use the terminology “phase of a point” as the image value, e.g., if  $\varphi(y) = x$ , then  $x$  is the “phase” of  $y$ . We will be explicit about the domain and codomain of such functions to avoid confusion.

## 2.3 A Motivating Simulation

### 2.3.1 Problem Statement

We begin our study with a motivating simulation example for a well-studied simple hybrid dynamical system - the Clock-Torqued Spring Loaded Inverted Pendulum (CT-SLIP) ([Siepel and Holmes, 2007](#)) model. As is apparent from the name, the CT-SLIP is an extension of the well-known SLIP model ([Blickhan, 1989b](#); [Seyfarth et al., 2002](#); [Ghigliazza et al., 2004](#); [Full and Koditschek, 1999](#)). We customized the spring force law to be a non-conservative Hill-like muscle model shown in (2.2), where the original SLIP had a Hookean spring. The Hill muscle model ([Hill, 1938](#)) has been postulated to be of sufficient accuracy to be useful for simulating human musculoskeletal behavior ([Winters, 1990](#); [Bogert et al., 1998](#))

$$F(\zeta) = K(L - \zeta)(1 - \eta\dot{\zeta}) - \mu\dot{\zeta} \quad (2.2)$$

The length  $L$  represents the length of the leg at touchdown. The parameters  $K$  and  $\eta$  provide averaged approximations to the length dependent and velocity dependent terms (respectively) of the Hill muscle model;  $\mu$  adds some dissipation, capturing the overall energy consuming nature of the task. We elected to employ it over a conservative spring as we want the freedom to inject or remove energy from the system to broaden the set of achievable motions.

Table 2.1: Parameters

Parameter	Definition	Nominal Value
$\eta$	$F_V$ average slope	0.03
$\mu$	dissipative loss	0.3
$L$	$F_L$ average slope	80
$t_s$	torsional spring at hip	0.1

The CT-SLIP has two legs which rotate around the center-of-mass (CoM). The

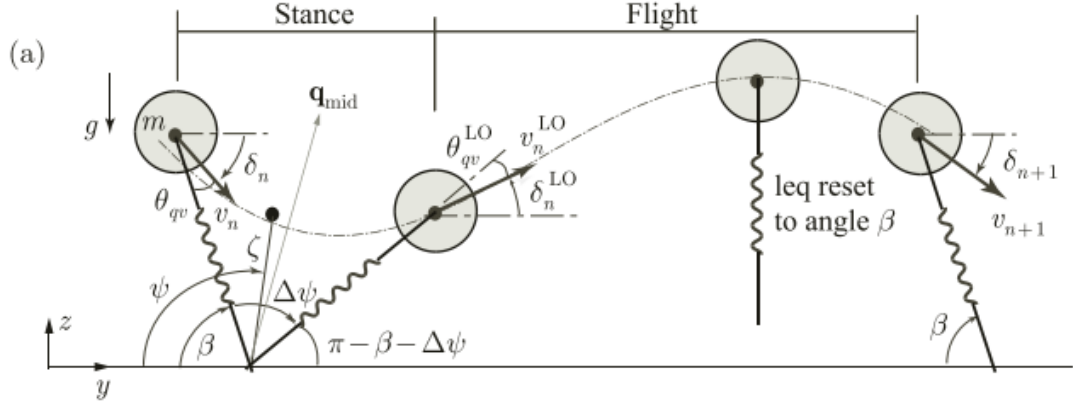


Figure 2.1: The CT-SLIP with indicating variables. For clarity, only one leg is shown. A periodic gait is an concatenation of a sequence trajectory segments from holonomic subsystems. Reproduced from Fig 2.2 of [Siepel and Holmes \(2007\)](#)

angle of each leg is determined by a piece-wise smooth feedforward (hence, “clocked”) reference curve that depends only on time - the Buehler clock ([Ghigliazza et al., 2004](#); [Saranli et al., 2001](#)). There is a torsional spring and actuator at the hip that tracks the reference signal with standard PID control upon angle.

The cycling legs, tuned for appropriate parameters, produce a stable forward motion of the CoM. We operationally assume without proof that a phase-like quantity exists for the CT-SLIP, and we use the numerical tool *Phaser* ([Revzen and Guckenheimer, 2008](#)) (which estimates phase from trajectory data) to produce a workable phase-map from hybrid data. See Appendix B for details on how phase should be defined for a hybrid model.

In this regime, we aim to recover a stable limit-cycle  $\Gamma$  post-damage. “Damage” is modeled as a destructive and irreversible parametric shift that causes the majority of trajectories from uniformly sampled initial conditions to crash (hip-mass striking ground). We will restrict our attention to the stance dynamics, as the aerial dynamics are ballistic – if we recover a desired motion in the stance phases, the aerial phases will be preserved as well.

For  $z = (x, y, \dot{x}, \dot{y}) \in TQ$ , let  $\dot{z} = f(z, \lambda, u)$  be the resulting Euler-Lagrange equa-

tions of the CT-SLIP hybrid model (see [Siepel and Holmes \(2007\)](#) for full equations of motion). Assume for parameters, described in table 2.1,  $\lambda_0 := (t_s^0, L^0, \mu^0, \eta^0) \in \mathcal{P} \subset \mathbb{R}^4$ , a stable periodic  $\Gamma$  exists, with stability basin  $\mathcal{B}$ . We conflate the geometric object  $\Gamma$  with an arbitrary solution on it parameterized by  $x_\gamma(t)$ .

Define the damaged system by setting  $\lambda_1 = (t_s^1, L^0, \mu^0, \eta^0)$ , where  $t_s^1 = t_s^0 + \delta$ . We assume that perturbation  $\delta$  is neither small in magnitude, nor reversible. That is, the damaged system has its torsional spring gain stuck at the value  $t_s^1$ , and that this value has destabilized  $\Gamma$ .

In Cartesian coordinates  $(x, y, \dot{x}, \dot{y})$  are the coordinates for the state of the CoM), we denote the cycle  $\Gamma$  as

$$\gamma_0(t) := (x_0(t), y_0(t), \dot{x}_0(t), \dot{y}_0(t)) \subset TQ, \quad t \in [0, T]$$

I.e.,  $\text{Im}(\gamma_0) = \Gamma$ . For  $\lambda_0$ , this is a stable limit cycle with phase map  $\varphi : \mathcal{B} \rightarrow \text{Im}(\gamma_0) = \Gamma$ .  $\varphi$  defines a diffeomorphism; for details, see Appendix B.

We propose the following design problem. Let  $E \in \mathcal{C}^\infty(TQ \times \mathcal{P}, \mathbb{R})$ . We use the notation  $z_\lambda$  for a trajectory with parameter set  $\lambda$ . Assuming that  $z(0) = z_0$ , find a  $\lambda_1$  such that the following two equations hold  $\forall t \in [0, T]$ .

$$\frac{d}{dt}\varphi(z_{\lambda_1}) = 1 \tag{2.3a}$$

$$\frac{d}{dt}E(\varphi(z_{\lambda_1})) = \frac{d}{dt}E(z_{\lambda_0}) \iff L_{f(z, \lambda_0)}(E \circ \varphi) = L_{f(z, \lambda_0)}E \tag{2.3b}$$

Eqn. (2.3) yields two equations on the four dimensional state space  $TQ$ . We could in theory solve this system pointwise for pairs  $(z, v) \in TQ$ . However, we want  $(z, v)$  to be an integrable submanifold, i.e., we want  $\frac{d}{dt}z = v$ . Including the two additional constraints that  $\frac{d}{dt}x = \dot{x}$ , and  $\frac{d}{dt}y = \dot{y}$ , we have a fully determined set of equations on  $TQ$ , so that a unique curve  $z(t)$  obeys them. (We will omit repeating these last two constraints for the remainder of the section for brevity, as they are straightforward.

They are merely insisting that we have a curve  $z(t)$  that obeys the constraints) That is, at a point  $x \in Q$ , there is a unique  $v \in T_x Q$  that satisfies Eqn. (2.3). Thus, if  $z(0) = \gamma_0(0)$ ,  $z(t) = \gamma_0(t)$ , satisfying our objective.

It is important to note that  $\frac{d}{dt}E(z_{\lambda_0})$  is taking the derivative of  $E$  *along the periodic orbit*, while  $\frac{d}{dt}E(\varphi(z_{\lambda_1}))$  is projecting a state  $x_{\lambda_1}$  that is *off* the orbit to its in-phase companion on the orbit, *then* differentiating. Equivalently, our goal is to modify  $\frac{d}{dt}E(\varphi(z_{\lambda_1}))$  so as to match the fixed reference signal  $\frac{d}{dt}E(z_{\lambda_0})$ .

The relationship is concisely expressed geometrically. Fix  $z(t) = z_0$ . Let  $r(t) = \frac{d}{dt}E(z_{\lambda_0}) = dE_{\varphi(z)}(D\varphi_z \cdot f(z, \lambda_0))$ . Let  $\omega, dE \in T^*\Gamma$ . Design  $\lambda_1 \in \mathcal{P}$ , subject to  $t_s = t_s^0 + \delta$ , such that, for  $t \in [0, T]$ ,

$$\omega_z(f(z, \lambda_1)) = 1 \quad (2.4a)$$

$$dE_{\varphi(z)}(D\varphi_z f(z, \lambda_1)) = r(t) \quad (2.4b)$$

The observant reader will notice this condition as *lifting* the constraints from  $\gamma_0$  to the neighborhood where  $\varphi$  (and thus  $\omega$ ) is defined. We are able to equate this condition on differential forms to signals of time by evaluating them along a specific trajectory. We will expand on this in the sequel.

### 2.3.2 Optimization

The objective is to fit the Lie derivatives shown in Eqn. (2.3) using measured/simulated trajectory data, assuming the that underlying vector field(s) is unknown. Since we are seeking a parameter  $\lambda_1$  with the irrevocable condition that  $t_s = t_s^0 + \delta$  that satisfies our tracking requirements, we could rephrase the design into a regression to *best fit* the constraints. I.e., for  $t \in [0, T]$ ,  $z_\lambda(0) = x_0$ ,  $\forall \lambda \in \mathcal{P} \mid_{t_s=t_s^1}$ ,

$$\lambda_1^* = \arg \min_{\mathcal{P}|_{t_s=t_s^1}} \left( \|\varphi(z_\lambda(t)) - \varphi(x_{\lambda_0}(t))\|^2 + \left\| \frac{d}{dt} (E \circ \varphi(z_\lambda(t))) - r(t) \right\|^2 \right)$$

The norm  $\|\cdot\|$  in the above is the  $L^2$  norm of functions

$$\|f\| = \left( \int_D |f|^2 \right)^{\frac{1}{2}} \quad (2.5)$$

Numerically, Eqn. (2.5) is approximated with the Euclidean 2-norm of  $\{f(x_i)\}_{i=1}^n \subset \mathbb{R}^n$ , where  $f(z_i)$  are sample values of  $f$  at the states  $z_i$  that are produced by the numerical integrator. Thus  $\lambda_1^*$  is the parameter that minimizes the largest absolute difference over the entire domain of  $t$ . Other function space norms can be selected, but as they are not generally equivalent, the value of  $\lambda_1^*$  depends on the choice. The requirement that  $\frac{d}{dt}z = v$  will be automatically enforced by the numerical integrator, so it does not need to explicitly be included in the cost function.

We elect to further modify our constraints. If we solve Eqn. (2.3) perfectly, we would have  $z_{\lambda_1}(t) = z_{\lambda_0}(t)$ . We instead relax (2.17) to instead be, for some constant  $c \in \mathbb{R}_{>0}$ ,

$$\omega(f(z, \lambda_1)) = c > 0 \quad (2.6)$$

By doing this, the phase rate of  $z_{\lambda_1^*}$  is not required to match that of  $\gamma_0$ , but merely be positive and constant. Geometrically, this means it must permute the isochrons in the same order, but not necessarily at the same rate.

Physically, this allows the damaged system to potentially have a different frequency than the undamaged system, yet the limb touchdown sequence is preserved. This choice was arbitrary on our part, but is motivated by the expectation that different spring parameters will have different resonant frequencies, so exactly matching the timing characteristics of the original cycle may not be possible.

### 2.3.3 Simulation Results

The CT-SLIP, and optimization problem of Eqn. (2.5), was numerically simulated in Python 2.7.5 using the NumPy and SciPy open-source numerical libraries.

We take the function  $E$  to be the elastic energy stored in the legs. Our optimization algorithm of choice is the Nelder-Mead implementation provided by the SciPy `optimize` library. We chose this method as it relies exclusively on zero-order terms of the cost function, limiting the computational burden required by each iterate. Our perspective is that we are attempting to recover from damage under conditions where model information is poor and expensive to determine; Nelder-Mead requires no knowledge except function evaluations. As we will see, despite the relatively simple optimization scheme, we obtain good performance.

Control of the CT-SLIP dynamics is accomplished parametrically - control inputs are restricted to manipulating parameters  $L, \mu$  and  $\eta$ , as  $t_s$  is fixed at its perturbed value. Each execution of the system has a fixed collection of parameters. Feedback is not being used to modify the system dynamics as a function of state. Rather, the optimization problem solving for fixed values of the parameters whose resultant dynamics satisfy the phase and energy conditions.

We additionally define the “total” energy  $E_T$  to be the sum of kinetic and potential energy of the CoM. The functions  $E$  and  $E_T$  were defined on the entire time domain by taking a Fourier series ( $n=15$ ) approximation of discrete values at sample points produced by numerical integration on the dynamics.

Since recovering the specific  $z_0(t)$  is not equivalent to having a *stable* controller, we elected to simultaneously fit an ensemble of trajectories emanating from a collection of randomly chosen initial conditions with radius  $r$  of  $z_0(0)$ . An ensemble of goal trajectories simulated at random initial conditions is generated as a ground truth to match, from which phase and derivatives can be determined without knowledge of the vector field. The randomly generated initial conditions are fixed at initialization;

they do not vary between function calls of the optimizer. The implementation of (2.3) subject to the relaxed phase constraint given in Eqn. (2.6) was encoded with the following cost function.

$$f(\lambda) := \left\| \frac{d}{dt} \left( \frac{E(\varphi(z_{\lambda_1}))}{\langle E_T(z_\lambda) \rangle} - \frac{E(z_{\lambda_0})}{\langle E_T(z_{\lambda_0}) \rangle} \right) \right\|^2 + \alpha \left\| \frac{\Delta\varphi}{\Delta t} - \langle \frac{\Delta\varphi}{\Delta t} \rangle \right\|^2 + \beta \left\| \langle \frac{\Delta\varphi}{\Delta t} \rangle \right\|^{-1} \quad (2.7)$$

$\langle \cdot \rangle$  denotes the mean along a sample path. The mean is used to normalize the terms as a proportion of the total average energy so that the same modulation is preserved, rather than attempting to enforce a particular absolute energy level.

The second term is the variance of the time derivative of phase along sample paths. Driving it to zero requires  $\varphi$  be constant along solutions. The last term is the inverse of the mean time derivative of phase which penalizes the frequency approaching zero. The last two terms are attempting for force  $\frac{\Delta\varphi}{\Delta t}$  to be a constant bounded away from 0, i.e. exactly Eqn. (2.6).  $\alpha, \beta \in \mathbb{R}$  are weighting coefficients that we do not argue how to select in a principled manner. For the purposes of our simulation, we chose  $\alpha = \beta = 1$ .

Shown in Fig. 2.2a and Fig. 2.2b are integration results for two initial conditions (out of 10) that were used to generate the fitting ensemble. For the goal system, all ten initial conditions stabilized to a periodic solution. For the perturbed system, seven out of ten initial conditions lead to a crash (wherein the CoM impacts the ground, and the simulation ceased). For the recovered solutions, nine out of ten initial conditions recovered to a periodic solution.

We see that the recovered system *on average* has superior performance over the perturbed system, but that the recovered system still does not match the unperturbed system's performance. The reason for the failure to fully recover is unknown, but we see that using differential constraints as output variables improves the performance of the system. Additionally, the output signal ( $r(t)$  in the above, for example) is defined



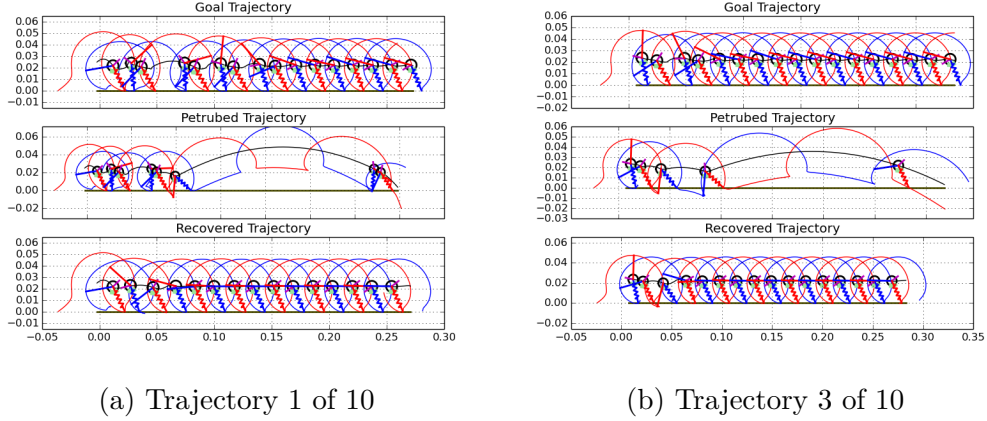


Figure 2.2: Traces of the CoM and legs. The CoM is black, one leg is red, and the other is blue. Ten initial conditions in the stability basin of the original periodic orbit were sampled. For the goal system, all were stable (top). When perturbed, 8/10 crashed (middle). The recovery method attempted to find a single  $\lambda_1^*$  such that all were stable. It succeeded only for 8/10 initial conditions.

by the constraints, rather than selected ahead of time, as one would in the case of output tracking.

We now build our observations from the CT-SLIP into a more general strategy.

## 2.4 Theory

We have seen in §2.3 that the constraint equations in Eqn. (2.3) provide a complete description of a desired limit cycle. By enforcing them, a specified trajectory was produced from the CT-SLIP. However, there are certainly unanswered questions. Does asymptotic phase help in some special way? What properties do the constraints need to satisfy? What if the control is not parametric?

We now present a class of mechanical systems relevant to locomotion that provably have a similar property, and furthermore, the constraints will always be on  $\mathbf{T}Q$ , rather than on  $\mathbf{TT}Q$ . In other words, we will show that there exist Pfaffian-like constraints that we can impose between position and velocity to restore a behavior. A significant feature of doing so is that the trajectories of mechanical systems have  $\mathbf{T}Q$  as their

state space. By expressing our objective on  $\mathbf{T}Q$ , it is independent of any underlying vector field. In other words, our approach generates a control objective that is agnostic to the specific dynamics that govern a robot.

### 2.4.1 Geometric Mechanics

We now present a brief overview of geometric mechanics and its relevance to locomotion. A considerable body of literature exists on this topic – the interested reader should consult [Bloch et al. \(1996\)](#); [Ostrowski \(1996\)](#); [Bloch et al. \(2003\)](#) and the references therein for precise details.

Assume that we have configuration space  $Q = S \times G$ , where  $G$  is a Lie group that left acts on  $Q$  freely and properly via  $\Phi_g$ , along with independent Pfaffian constraints  $\omega(q) \cdot \dot{q} = 0 \in \mathbb{R}^{k1}$ , and Lagrangian  $L : \mathbf{T}Q \rightarrow \mathbb{R}$ . We critically assume that both  $L$  and the constraints  $\omega$  are *symmetric under  $G$* , i.e.

$$L(q, \dot{q}) = L(\phi_g(q), D_q \phi_g \dot{q}) \quad (2.8)$$

$$\omega(q) \cdot \dot{q} = D_q \phi_g^T \cdot \omega_{\phi_g(q)} \cdot \dot{q} \quad (2.9)$$

If  $k$ , the number of constraints, is equal to  $\dim(S)$ , it can be shown ([Ostrowski and Burdick, 1998](#); [Bloch et al., 2003, 1996](#); [Koon and Marsden, 1997](#); [Hatton and Choset, 2011](#)). that there exists a map  $A : \mathbf{T}S \rightarrow \mathfrak{g}$  such that

$$g^{-1} \dot{g} = -A(s) \cdot \dot{s} \quad (2.10)$$

Thus, the body velocity  $g^{-1} \dot{g}$ <sup>2</sup> is explicitly a function of shape  $s$  and shape velocity  $\dot{s}$ . Physically, for  $k = \dim(S)$ , then the system is kinematic - the motion of the robot

---

<sup>1</sup>The affine case  $\omega(q) \cdot \dot{q} = \gamma(q)$  is also covered in the listed references

<sup>2</sup>by which we mean  $g^{-1} \dot{g} := \xi = T_g L_{g^{-1}} \dot{g}$ , which is the velocity of the body frame in body coordinates.

is completely determined via shape, with no drift due to momentum.

If  $k < \dim S$ , there is a generalized momentum term  $p$  that appears affinely in the reconstruction equation ([Bloch et al., 1996](#)).

$$g^{-1}\dot{g} = -A(s) \cdot \dot{s} + B(s)p \quad (2.11)$$

There is an associated equation  $\dot{p} = C(r, \dot{r}, p)$  that determines how the momentum terms evolve, and accounts for all the “second order” features of the mechanical system.

Consider curves  $s(t) \subset S$  that are periodic with period  $T$ . Eqn. (2.11) is a differential equation on the group  $G$  that defines the *holonomy* or *horizontal lift* ([Kobayashi and Nomizu, 1963](#)) of a path  $s(t), t \in [0, T] \in S$ . While the dynamics of  $s$  are unimportant to validity of this representation, we will always assume that the dynamics of  $s$  are controlled, i.e. there exists some equation

$$\dot{s} = f(s) + G(s)u$$

That is, the shape variables evolve as a first-order control-affine system, but there are sufficient constraints that there is an explicit relationship between the body velocity and shape state at a given time.<sup>3</sup> A solution curve  $c(t) := (s(t), g(t))$  is called *horizontal* by virtue of satisfying  $\omega(c(t)) \cdot \dot{c}(t) = 0$ . The curve  $c(t)$  is a path connecting  $g(0)$  to  $g(T)$  that satisfies the constraints, with holonomy  $g(T)g(0)^{-1}$ . The path  $c$  is not necessarily unique – the set of all curves that achieve a desired group motion is a Lie group in its own right ([Kobayashi and Nomizu, 1963](#)).

$$\text{Hol}(m) = \{g_s \mid s : [0, T] \rightarrow S, s(0) = s(1)\} \quad (2.12)$$

---

<sup>3</sup>Some authors take it a step further, and simply have  $\ddot{s} = u$ .

Solutions are generally difficult to determine, and can be expressed by various expansions, such as a path-ordered exponential ([Marsden et al., 1990](#)) or Magnus expansion ([Radford and Burdick, 1998](#)). In the case that the group  $G$  is abelian, conventional integration suffices ([Marsden et al., 1990](#))

$$g(t) = \exp \left( \int_0^t \xi(s) ds \right)$$

Classic kinematic examples would include the kinematic car ([Pepy et al., 2006](#)) or Purcell swimmer ([Hatton and Choset, 2013](#)); a classic non-kinematic example is the snakeboard ([Bloch et al., 1996](#)).

The group motion  $g$  is the displacement in the world, while the  $s$  (shape) variables describe relative motion of the robot limbs with respect to the center of mass. For a closed loop  $s(t), t \in [0, T]$ , a single revolution produces a holonomy element  $h := g(T)g(0)^{-1} \in G$ . In mechanics, this is referred to as the *geometric phase* ([Marsden et al., 1990](#)) of the loop  $s(t)$ .<sup>4</sup> Repeated iterations in  $S$  simply concatenate the group elements  $h$  owing to its group structure. In this sense, locomotion is well represented. Repeated cycling of limb motions produces displacement in the world, agnostic to starting position. In general, the same group motion can be effected by many gaits.

The point we wish to take away from the preceding discussion is that the constraints  $\omega$  introduce a structure that allows the “output” variables  $g$  to be written as a function of “input” variables  $s, \dot{s}$ , and that this structure depends *only on the constraints* when  $k = \dim G$ , rather than on  $L$  as well.

### 2.4.2 Something Obvious

To motivate the construction we present in the sequel, consider the following problem. Let  $N$  be a manifold of dimension  $n$ , and assume there exists flow  $\phi^t$  :

---

<sup>4</sup>When the affine influence of  $p$  is non-zero, there is a drift-like term called the *dynamic phase* ([Marsden et al., 1990](#)) - the product of the geometric and dynamic phases is the total group element that results from a closed path  $s(t)$  being executed in the shape variables.

$N \rightarrow N$ , along with  $C^1$  function  $F : N \rightarrow \mathbb{R}^k$ . Suppose that we have initial condition  $y(0) \in N$  such that  $F(\phi^t(0)) = 0 \in \mathbb{R}^k, \forall t \in [a, b]$ . In a local coordinates, we have  $\phi^t(y(0)) =: (y_1(t), \dots, y_n(t))$ ,  $F(y_1(t), \dots, y_n(t)) = 0$ . Assume that  $F$  is full rank at each point  $y(t)$ , so that by the implicit function theorem ([Lee, 2013](#)), there exists  $C^1$  function  $h$  such that

$$F(h(y_{k+1}, \dots, y_n), y_{k+1}, \dots, y_n) = 0$$

Assume now that there exists a submersion  $\varphi : M \rightarrow N$ . We may then consider the pullback  $\varphi^*h(m) := h(\varphi(m)_{k+1}, \dots, \varphi(m)_n)$ . Suppose we found a curve  $m(t) \subset M$  such that

$$\varphi^*h(m(t)) = (y(t)_{k+1}, \dots, y(t)_n) \quad (2.13)$$

and

$$(\varphi(m(t))_{k+1}, \dots, \varphi(m(t))_n) = (y(t)_{k+1}, \dots, y(t)_n) \quad (2.14)$$

Then,

$$F(\varphi(m(t))) = 0 \quad (2.15)$$

While the above is a direct application of the Implicit Function Theorem ([Lee, 2013](#), Thm. C.40), it expresses the design problem of finding  $m(t)$  as two testable objectives. First, insist that  $\varphi(m(t))$ , the “phase” of  $m(t)$ , is a specified reference, and second, that  $m(t)$  satisfies the constraint defined Eqn. (2.13). As neither the kernel of  $\varphi$  or  $\varphi^*h$  is non-trivial, there are many possible equivalent candidates for  $m(t)$ . If an initial solution curve  $m_1(t) \subset M$  is no longer achievable due to damage, we can find for another curve  $m_2(t)$  that *exactly* solves  $F = 0$ . We now apply this idea to our problem in geometric mechanics.

### 2.4.3 The Kinematic Connection Case

Following our approach in 2.4.1, let  $Q = S \times G$  for Lie group  $G$ , with  $G$ -invariant Lagrangian  $L : \mathbf{T}Q \rightarrow \mathbb{R}$ . All functions will be assumed to be  $C^\infty$  unless otherwise stated. Assume that we are equipped with Pfaffian constraint form  $\omega \in (\Omega^1)^k$ , i.e, a collection of  $k$  linearly independent 1-forms  $\omega_i, i = 1, \dots, k$  such that  $\omega_i(q)\dot{q} = 0$  for all solutions  $(q, \dot{q})$  determined by  $L$ . We have that  $\omega$  is  $G$ -equivariant. In coordinates, this implies we may write it as, for  $\dot{q} = (\dot{r}, \xi = g^{-1}\dot{g})$  :

$$\omega_\xi(r)\xi + \omega_{\dot{r}}(r)\dot{r} = 0 \in \mathbb{R}^k$$

We assume that  $k = \dim(G)$ , so that we are in the principal kinematic case. Assume  $\dim(S) = n > k$ , so for  $\omega_\xi \in GL(k, k)$ , and  $\omega_{\dot{r}} \in L(n, k)$ , where  $L(n, k)$  is the space of linear maps from  $\mathbb{R}^n \rightarrow \mathbb{R}^k$ . Then we can define map  $A$  by:

$$-\omega_\xi^{-1}\omega_{\dot{r}} \cdot \dot{r} =: -A(r) \cdot \dot{r} = \xi$$

#### 2.4.3.1 Encoding Templates and Recovery

Let  $\hat{Q} = \hat{S} \times G$ , and assume that we have submersion  $\varphi : \hat{Q} \rightarrow Q$  such that  $\varphi(\hat{s}, g) = (\varphi^s(\hat{s}), g)$ , i.e. it is the identity map on  $G$ . We then have a map  $\varphi^s : \hat{S} \rightarrow S$ , which is also full rank. Given the constraint form  $\omega : \mathbf{T}S \times \mathfrak{g} \rightarrow \mathbb{R}^k$ , we may pull back to  $\varphi^*\omega(\hat{s}) \cdot (\dot{\hat{s}}, \xi) = \omega(\varphi^s(\hat{s})) \cdot D\varphi^s \cdot (\dot{\hat{s}}, \xi)$ . We assume that we are given curve  $q(t) = (s(t), g(t)) \subset Q$  such that  $\omega(q) \cdot \dot{q} = 0$ .

Suppose that we had curve  $\hat{q}(t) = (\hat{s}(t), \tilde{g}(t)) \subset \hat{Q}$  such that, f or  $v(t) := \varphi^{\hat{s}}$ ,

$$v(t) = s(t) \tag{2.16}$$

and that

$$\varphi^* \omega(\hat{s}) \cdot (\dot{\hat{s}}, \tilde{\xi}) = 0. \quad (2.17)$$

Eqn. (2.17), at each point  $\hat{s}$ , can be thought of as  $\dim G$  linear equations in  $\dim Q \times \dim G$  unknowns. By assuming Eqn. (2.16) holds, we also have

$$\frac{d}{dt}v(t) = \frac{d}{dt}s(t)$$

Ergo, along  $v(t)$  and  $s(t)$ ,  $\omega(v(t), \cdot) = \omega(s(t), \cdot)$ . Thus, we have  $\dim G$ -equations in  $\dim G$  unknowns. We initially assumed the problem had rank  $\dim G$ , so that there is a unique solution, so, for  $\tilde{\xi} = \tilde{g}^{-1} \dot{\tilde{g}}$ ,

$$\omega(s, \xi) = 0 = \omega(v, \tilde{\xi}) \implies \xi = \tilde{\xi}. \quad (2.18)$$

We call the codomain  $Q$  for phase map  $\varphi$  the *encoding template*. It represents a simpler constrained system (as we assume  $\dim Q < \dim \hat{Q}$ ) that is still capable of executing the desired  $\xi(t)$ .

Thus, our recovery strategy consists of two phases. An initial training phase on the undamaged system, where the template  $Q$ , map  $\varphi$ , and constraints  $\omega$  are defined. Assume that the working robot was executing curve  $x(t)$  that yields the holonomy of  $g(t)$ ; by definition, part of our desired behavior is  $g(t)$ , so that any curve with the same holonomy is functionally equivalent. Then, we project this down to  $s(t)$  via  $\varphi$ , obtaining the R.H.S of (2.16). By assumption, we have set  $\omega$  such that (2.17) is satisfied. While this assumption is strong, we will relax it considerably in the sequel.

Then, for a damaged robot, we design a new  $\hat{x}(t)$  such that Eqn. (2.16) and Eqn. (2.17) are both satisfied. As we see from Eqn. (2.18), this implies the holonomy  $g(t)$  is achieved, despite that  $x(t) \neq \hat{x}(t)$  necessarily.

An important feature of the system on  $Q$  is that we *only* require the above re-

relationships for our preferred  $g(t)$ . The template system may be able to engage in behaviors that the anchor system is *not* able to do - not all trajectories can or need to be lifted to trajectories of the anchor system.

Our assumptions in this section correspond to identifying an encoding template  $Q$  that has the same CoM as the robot, for example, if we had a six-legged robot that moved in the plane with configuration  $\hat{Q} \times SE(2)$ , we could have an encoding template  $Q \times SE(2)$  with constraints of a kinematic car whose center-of-mass coincided with the robot's center-of-mass for all time. Then, by recovering the chosen trajectory of the kinematic car, the group motion of the anchor robot is also preserved, even though the robot's limb motions may be very different from their original behavior.

#### 2.4.3.2 Missing Constraints

Assume that we have a known embedded curve  $(r_0(t), g_0(t)) \subset S \times G$ ,  $t \in (t_a, t_b)$  such that  $\xi_0(t) := g_0^{-1}(t)\dot{g}_0(t) = A(r_0(t)) \cdot \dot{r}_0(t)$ . Since we have a “wide” matrix  $A$ , there are infinitely many curves  $r(t)$  that map to  $\xi_0(t)$ , as by fiat we assume there exists at least one  $r_0$ . For  $\Gamma := \text{Im}(r_0(t), g_0(t))$ , assume that  $\text{rank}(A|_{\pi_1 \circ \Gamma}) = s$  is constant. Since rank is an open condition, there exists a tubular neighborhood  $U \subset S$  of  $r_0(t)$  s.t.  $\text{rank}(A|_U) = s$  is constant.

Let  $f(r, g) : S \times G \rightarrow \mathbb{R}$  be a real-valued function, then define

$$\eta(t) := f(r_0(t), g_0(t))$$

have retraction  $\varphi : U \rightarrow \Gamma$ , and define the pullback

$$\tilde{f}(r, g) = f \circ \varphi \tag{2.19}$$

so that by definition,

$$\tilde{f}|_{\Gamma} = f \tag{2.20}$$



Implicit in the definition of  $\varphi$  is the time-parameterization of  $(r_0, \xi_0)$ . Explicitly,  $\theta(t) = (r_0(t), g_0(t))$  is the embedding of  $(t_a, t_b)$

$$\tilde{f}(r, g) = f(\pi_1 \circ \varphi(r, g), \pi_2 \circ \varphi(r, g)) = \eta(\theta^{-1} \circ \varphi(r, g)) \quad (2.21)$$

$\pi_1 : S \times G \rightarrow S$ , and  $\pi_2 : S \times G \rightarrow G$ . We write via chain rule, on  $\Gamma$

$$d\tilde{f} \cdot (\dot{r}_0(t), \dot{g}_0(t)) = df \cdot (\dot{r}_0(t), \dot{g}_0(t)) \quad (2.22)$$

$$= df \cdot (\dot{r}_0(\theta^{-1}((r_0, g_0))), \dot{g}_0(\theta^{-1}((r_0, g_0)))) \quad (2.23)$$

$$=: \gamma(r_0, g_0) \quad (2.24)$$

Since  $\tilde{f}$  is defined everywhere on  $U$ , so is its derivative  $d\tilde{f}$ . This allows us to formulate the constraint

$$d\tilde{f} \cdot (\dot{r}, \dot{g}) = df \circ d\varphi \cdot (\dot{r}, \dot{g}) = \gamma(\varphi(r, g)) \quad (2.25)$$

Assume now that the function  $\varphi(r, g)$  is group-invariant, and that  $f$  generates a foliation of  $Q$  that is permuted by  $\phi_h$ , so that

$$\tilde{f}(r, g) = \tilde{f}(r, \phi_h(g)) + c(h)$$

Where  $c : G \rightarrow \mathbb{R}$  is some function as smooth as  $f$ . Differentially, this implies  $\forall r \in S, g, h \in G$ , where  $d\tilde{f} = \partial_r \tilde{f}(r, g)dr + \partial_g \tilde{f}dg$

$$\partial_r \tilde{f}(r, g)dr = \partial_r \tilde{f}(r, \phi_h(g))dr \quad (2.26)$$

$$\partial_g \tilde{f}(r, g)dg = \partial_g \tilde{f}(r, \phi_h(g))dg \cdot d\phi_h \quad (2.27)$$

unique

The first equation implies  $\partial_r \tilde{f}$  does not depend on  $g$ . The second equation implies, for  $h = g^{-1}$ ,

$$\partial_g \tilde{f}(r, g) = \partial_g \tilde{f}(r, e) d\phi_{g^{-1}} = \partial_g \tilde{f}(r, e) \cdot g^{-1}$$

where in the last statement we have conflated the action and lifted action of  $G$ .

These together let us write

$$\partial_r \tilde{f}(r) dr + \partial_g \tilde{f}(r, e) dg \cdot g^{-1} = \gamma(r) \quad (2.28)$$

We have  $\gamma(r)$  as by assumption  $\varphi(r, hg) = \varphi(r, g)$ , implying it is only a function of  $r$ . We simplify notation and write (2.28) as, for  $\xi = g^{-1} \dot{g}$ , and implicitly assuming that it is along curve  $(r(t), g(t))$

$$\tilde{\omega}_r(r) \dot{r} + \tilde{\omega}_\xi(r) \xi = \gamma(r) \quad (2.29)$$

We now assume that our robot has been damaged in a way that alters the set of active constraints in a low-rank way. Our fundamental assumption is that the damage can be modeled as that there  $\exists i \in \{1, \dots, k\}$  such that the known constraint  $\omega_i$  is no longer in effect - i.e. solutions curves are not required to satisfy it.

Since the 1-form determined by (2.28) is group-equivariant, we can consider the quantity,  $i = 1, \dots, k - 1$

$$\omega_{1\xi} \xi + \omega_{1\dot{r}} \dot{r} = 0 \quad (2.30)$$

$$\vdots$$

$$\omega_{(k-1)\xi} \xi + \omega_{(k-1)\dot{r}} \dot{r} = 0 \quad (2.31)$$

$$\tilde{\omega}_r(r) + \tilde{\omega}_\xi \xi = \gamma(r) \quad (2.32)$$

where we have replace  $\omega_i$  with  $d\tilde{f} = \gamma(r)$ . Call this form  $\hat{\omega}$ . We have generated a

form by pulling back  $f$  into  $M$ . Observe  $\hat{\omega}$  is *still kinematic*, i.e. by obeying it, the group element  $\xi$  is enforced. However, for the implication in Eqn. (2.18) to hold, the rank of (2.29) needs to be  $\dim G$ .

### 2.4.3.3 Do we require kinematic systems?

An **key** aspect of §2.4.3.2 is that we no longer require  $\omega(q) \cdot \dot{q} = 0$ , but rather we allow *affine* constraints. In fact, we never needed to assume the constraints were Pfaffian as in 2.4.3, and could have started with affine constraints. However, we elected to begin with kinematic systems due to their relative familiarity.

In this sense, the reconstruction equation approach only requires that the dynamics are *fully* constrained in the group directions, rather than *kinematic* in the sense that they have no history – the “shape” variables could have arbitrary dynamics, but if there are enough constraints, there is still an *functional* relationship to the group motion.

Ergo, this approach works for any constrained mechanical system that is invariant under the action of a group, rather than only those that are properly *kinematic*. As we will see in the sequel, even this can be relaxed.

### 2.4.3.4 Transversality of Group Invariant Functions

We now show that completing the constraints with differentials such that the rank is expanded is a generic property. We use  $L^r(n, k)$  to denote the  $n \times k$  matrices that have corank  $r$ , and  $L(n, k) = \bigcup_r L^r(n, k)$ . Consider an  $k \times n$  matrix-valued function  $A : S \rightarrow \text{Hom}(V, W)$

$$A = \begin{bmatrix} a_1(s) \\ a_2(s) \\ \vdots \\ a_k(s) \end{bmatrix} \tag{2.33}$$

Each  $a_i, i \in 1, \dots, k$  denotes a row. Assume that  $A(s)$  is of rank  $k$  for all  $s \in S$ . Let  $f \in C^\infty(S, \text{Hom}(\mathbb{R}^n, \mathbb{R}^N))$  be a smooth function, for  $N \in \mathbb{N}$ ,  $N \geq 1$ , and define

$$B(s) := \begin{bmatrix} A(s) \\ f(s) \end{bmatrix} \quad (2.34)$$

I.e. we augment the rows of  $A$  with a  $N$  new rows determined by the function  $f$ . We would like to determine when our choice of  $f$  will have the rank of  $B \geq k + 1$ .

We make use of the following theorem.

**Theorem II.1** (Prop 5.3, ([Golubitsky and Guillemin, 2012](#))).  *$L^r(V, W)$  is a closed submanifold of  $\text{Hom}(V, W)$  with  $\text{codim}(L^r(V, W)) = (m - q + r)(n - q + r)$  where  $q = \min(m, n)$ .*

The dimension of  $B$  is  $(k + N) \times n$ . We have  $m = k + N, n = n, q = m, r = k + N - k = N$ . Since adding a row cannot lower the rank of a matrix, we equivalently want to ensure that  $B \notin L^N$ . So,  $\text{codim}(L^N) = N(n - k)$ .

Let  $X, Y$  be smooth manifolds, and  $f : X \rightarrow Y$ . Recall, for  $W$  an immersed submanifold of  $Y$ , and  $x \in X$ , the map  $f$  is transverse to  $W$  at  $x$  if either (a)  $f(x) \notin W$ , or (b)  $\mathbf{T}_{f(x)}Y = \mathbf{T}_{f(x)}W + (df)_x(\mathbf{T}_xX)$ .

We now state a relatively apparent fact about transverse maps.

**Theorem II.2** (Prop 4.2, ([Golubitsky and Guillemin, 2012](#))). *Let  $X$  and  $Y$  be smooth manifolds,  $W \subset Y$  a submanifold. Suppose  $\dim X + \dim W \leq \dim Y$ . (i.e.  $-\dim X \leq \text{codim}(W)$ ) Let  $f : X \rightarrow Y$  be smooth and suppose that  $f$  is transverse to  $W$ . Then  $f(X) \cap W = \emptyset$ .*

So by Prop. II.2, a sufficient condition for  $B(s) \notin L^N$  for all  $s \in S$  is: Assume that  $B$  is transverse to  $L^N$ , and  $N(n - k) > n$ . If so, the only way for  $B$  to successfully transverse to  $L^N$  to have  $B(s) \notin L^N$  by definition, as  $\dim(\text{Im}(B)) = n$ ,  $\dim L^N =$

$N + k - N(n - k)$ . Ergo, if  $B$  is transverse

$$N > \frac{n}{n - k} \quad (2.35)$$

Then  $B(s)$  will have rank greater than  $k$ .

Now, we are “likely” to have  $B$  be transverse by the following.

**Theorem II.3** (Cor. 4.12, ([Golubitsky and Guillemin, 2012](#))). *Let  $X$  and  $Y$  be smooth manifolds, and let  $W$  be a submanifold of  $Y$ . Then, the set of smooth mappings of  $X$  to  $Y$  which intersect  $W$  transversely is dense in  $C^\infty(X, Y)$ , and if  $W$  is closed, then this set is also open.*

$L^N$  is closed, so by the above there is an open and dense set of maps  $f$  such that  $B$  will be rank greater than  $k$ .

We now modify the problem to ask a slightly different question. Suppose instead picking functions that directly provide rows of  $B$ , we wanted to pick  $N$  functions  $f \in C^\infty(S, \mathbb{R})$  such that the matrix-function extension,

$$B(s) := \begin{bmatrix} A(s) \\ df(s) \end{bmatrix} \quad (2.36)$$

had  $\text{rank}(B(s)) \geq k + 1$ ?

Let  $X$  and  $Y$  be two smooth manifolds, and let  $p \in X$ . Let  $f, g : X \rightarrow Y$  be smooth maps with  $f(p) = g(p) = q$ .  $f$  has first-order contact with  $g$  at  $p$  if  $(df)_p = (dg)_p$  as mappings of  $\mathbf{T}_p X \rightarrow \mathbf{T}_q Y$ , denoted via  $f \sim g$  at  $p$ .  $J^1(X, Y)_{(p, q)}$  is the set of equivalence classes under  $\sim$ . Let  $J^1(X, Y) := \bigcup_{(p, q) \in X \times Y} J^1(X, Y)_{(p, q)}$ ; it is a smooth manifold in its own right ([Golubitsky and Guillemin, 2012](#), Thm. 2.7).

Let  $\sigma \in J^1(X, Y)$ ; let  $f$  represent  $\sigma$ .  $df_p$  is a linear mapping. Define  $\text{rank}(\sigma) =$

$\text{rank}(df_p)$ . Define the set <sup>5</sup>

$$S_r := \{\sigma \in J^1(X, Y) \mid \text{corank}(\sigma) = r\}$$

Then we have the following theorem.

**Theorem II.4** (Prop 5.4, ([Golubitsky and Guillemin, 2012](#))).  *$S_r$  is a submanifold of  $J^1(X, Y)$  with  $\text{codim}(S_r) = (n - q + r)(m - q + r)$ . In fact,  $S_r$  is a subbundle of  $J^1(X, Y)$ , with fiber  $L^r(\mathbb{R}^n, \mathbb{R}^m)$*

If we take  $W = S_N$ , we want  $\sigma$  transverse to  $W$ . Since  $J^1(X, Y)$  is a manifold, and  $S_N$  is closed, we immediately see by the previous that the space of jets that are transverse to  $W$  is open and dense. Motivated by this observation, we select  $N > \frac{n}{n-k}$  differentiable functions to have rank greater than or equal to  $k + 1$ .

We have shown that we can add rank to a set of constraints by using the derivative of some  $f \in \mathcal{C}^\infty(S, \mathbb{R}^N)$ . However, for our problem, we want the  $f$  we use to have *group-invariant* constraints, like above, so that we still have a connection.

It is clear that if the level sets of  $f$  are permuted by  $\phi_h$ ,

$$f(r, \phi_h \cdot g) - f(r, g) = c(h) \tag{2.37}$$

Let  $g = e$ , so that

$$f(r, h) = c(h) + f(r, e) \tag{2.38}$$

$c \in \mathcal{C}^\infty(G, \mathbb{R})$ , and since  $e$  is fixed, we think of  $f_e(r) \in f(R, \mathbb{R}^n)$  as a pure function of shape. Ergo, let  $\hat{c} \in \mathcal{C}^\infty(G, \mathbb{R})$ ,  $\hat{f} \in f(S, \mathbb{R}^n)$ . We can define a function by

$$f(r, g) := \hat{c}(g) + \hat{f}(r) \tag{2.39}$$

---

<sup>5</sup> $\text{corank}(df)_p = \min(\dim X, \dim Y) - \text{rank}(df)_p$

To see that  $f(r, \phi_h \cdot g) - f(r, g) = c(h)$ , we compute

$$f(r, \phi_h \cdot g) = \hat{c}(\phi_h \cdot g) + \hat{f}(r) \quad (2.40)$$

$$f(r, \phi_h \cdot g) - f(r, g) = \hat{c}(\phi_h \cdot g) - \hat{c}(g) = \hat{d}(h) \quad (2.41)$$

Where  $\hat{d}$  is a function only of  $h$ . Thus, let  $c_f \in \mathcal{C}(G \times S, \mathbb{R})$ ,  $c_f(g, s) = c(g) + f(s)$ .

Since there is an open and dense set of functions  $f$  that provide the necessary rank condition, and we see that we can “complete”  $f$  to  $c_f$  by picking  $c$ , we see that there is an open and dense set of group invariant functions that meet our rank requirement.

#### 2.4.4 Generalization Away From Mechanics

Section §2.4.3 makes use of geometric mechanics, but upon closer inspection, is not strongly incumbent on the existence of a principal fiber bundle, Lagrangians, or even symmetry. The basic argument was that a sufficiently rich collection of constraints  $\omega$  is adequate to fully determine a desired curve  $(s(t), \xi(t))$  in the template. In this sense, even the presence of any dynamical structure at all is superfluous. We only care about the relationship between tangent vectors and base points, rather than solving for curves that obey them. Armed with this idea, we can remove the symmetry and mechanical structure as follows.

We now assume that a robot’s motion is determined by curves  $x(t)$  taking values in a manifold  $Q_R$  on which we can write differential constraints. Typical choices for  $Q_R$  could be the configuration space of the robot body (such as in the geometric mechanics case), its phase space, or a more general state space. While it may seem initially strange to allow the domain to vary so generally, we observe that a “tangent vector”, which is our fundamental object of interest, is naturally defined in an equally general setting.

As general systems have no natural equivalent to the group variables we preserved

in the geometric mechanics case, we instead define a *behavior specification* to be a list of constraints of the form  $\Omega_i(x) \cdot \dot{x} = \gamma_i(t, x)$ ,  $i = 1, \dots, k$ . Here again each  $\Omega_i$  is a differential 1-form. The vector  $\gamma(t, x) := (\gamma_1, \dots, \gamma_k)$  is a vector of length  $k$  that defines the value the constraint functions  $\Omega_i$  must satisfy, and therefore takes values in the same codomain as that of the 1-forms, i.e. generally  $\gamma \in \mathbb{R}^k$ . The list of  $\Omega_i$  contains any inviolate physical constraints that determine the physics of the robot, as well as constraints we as designers wish to engineer into the system. Together, we obtain constraints

$$\Omega(x) \cdot \dot{x} = \gamma(t, x) \quad (2.42)$$

where the rows of  $\Omega$  are the  $\Omega_i(x)$  in the standard convention. We assume the matrix  $\Omega$  to be of constant, though necessarily not full, rank when evaluated along admissible curves of  $Q_R$ . Formally speaking, the behavioral specification is the pair  $(\Omega, \gamma)$ .

A curve  $x(t)$  taking values in  $Q_R$  which satisfies Eqn. (2.42) is an instance of the behavior. A major feature of our behavior representation is that it is agnostic of the mechanism that generates curves  $x(t)$ . In particular, instances of the behavior may intersect and even overlap, only to diverge later – unlike trajectories of conventional closed loop control models. With respect to a behavior specification there are only curves that satisfy the constraints, and those that do not.

We chose this definition for a behavior as it contains a number of special cases. For example, if  $\Omega$  is invertible everywhere, we may write  $\dot{x} = \Omega^{-1}(x)\gamma(t, x)$  – a conventional non-autonomous ordinary differential equation (ODE). Here instances of the behavior are solutions of the ODE. The popularly used class of affine control systems  $\dot{x} = f(x) + G(x)u$  can be represented using a pseudoinverse  $G^\dagger$  of  $G$ , and constructing  $\Omega(x) := I - G(x)G^\dagger(x)$  and  $\gamma(t, x) := \Omega(x)f(x)$ . This is a standard application of control redesign. If the space  $Q_R$  is taken as the configuration space of a mechanical system, Pfaffian and affine differential constraints are behavior specifications as well. In kinematic reduction, the constraints would be the metric inner products of Eqn.



11 in [Bullo et al. \(2002\)](#). The preceding list of model types that can be realized as behavior specifications is not exhaustive, but is intended to indicate that a number of useful constructions of control and robotics are naturally encapsulated in our proposed definition.

While the constraints required by physics are intrinsic to the system, it is not immediately clear from the definition given of how to encode a design goal as a collection of constraints. In exactly the same manner as the CT-SLIP simulation, and the group-invariant case, we propose the following strategy: we first find a manifold  $Q_E$  and a full-rank function  $\varphi : Q_R \rightarrow Q_E$  such that we are certain that whatever outcome we desire is realized by a behavior specification on  $Q_E$ . The space  $Q_E$  is the generalized encoding template, e.g., in §2.4.3,  $Q$  was the encoding template for anchor  $\hat{Q}$ ; e.g.,  $Q_E = \Gamma$  for the CT-SLIP. As before, we take the dimension of  $Q_E$  to be less than that of  $Q_R$ . The map  $\varphi$  reduces the coordinates of  $Q_R$  to values we as designers care about encoding; for example,  $\varphi$  could return the CoM coordinates, an end effector location, joints angles, etc. The map  $\varphi$  can equally be considered a collection of *outputs*  $y_i := \varphi_i(x)$ ,  $i = 1, \dots, \dim Q_E$  when evaluated along a curve  $x(t)$ .

To encode a behavior, we write a behavior specification  $\omega_i(y) \cdot \dot{y} = \eta_i(t, y)$  on the output variables / encoding template. Such a construction precisely includes all the special cases a behavioral specification can capture, but only on the output variables. We can then pull the  $\omega_i$  back to  $Q_R$  to augment any extant  $\Omega$  by  $\Omega_j$ . In matrix form, pulling them back is merely adding rows  $\Omega_j(x) \cdot v := \omega_j(\phi(x)) \cdot D\phi(x) \cdot v$  to the matrix  $\Omega$ . We augment our notation to include  $\varphi$  in the “behavioral specification” as the tuple  $(\varphi, \Omega, \gamma)$ , indicating that it is the image of  $\varphi$  which is the target of our design efforts. This emphasizes that there are virtual constraints on the output variables in concert with pre-existing constraints that are defined only on  $Q_R$ .

As before, we assume that we started with a functional robot to evaluate our

constraints upon in a “training” stage. We assume that there was a distinguished curve  $x_0(t) \subset Q_R$  that satisfied a given behavior specification  $(\phi, \Omega, \gamma)$ . We will consider the case where the number of constraints *exceeds* the dimension of  $Q_R$ , but that the output constraints  $\omega$  are satisfied without additional control effort. For emphasis, the template constraints  $\omega$  *define* (conversely, are defined by)  $\eta$ , so they are satisfied by the working system by definition. E.g., compare Eqn. (2.4a) and Eqn. (2.16) - the constraint form evaluated on a desired motion defined the value of the constraint; control effort was not needed to enforce it. When the number of constraints exceeds the dimension of  $Q_R$ , the rows defined by  $\omega$  are necessarily linearly dependent with the rows of  $\Omega$  - the rank of  $\omega$  augmenting  $\Omega$  is identical to that of  $\Omega$  alone along desired trajectories in  $Q_R$ . As such, the constraints  $\omega$  are redundant for the working robot.

We assume then that the robot is disrupted in a manner to introduces a new  $\Omega_r$ , or eliminates one of the native  $\Omega_i$ , representing effects such as motors seizing, limbs breaking off, etc. The recovery strategy is to re-enforce, via control, the output constraints  $\omega_i$ , which are presumably violated by whatever motion the broken robot is performing without compensation. In this the  $\omega$  output constraints which were originally redundant now play the essential role of a more complete behavior specification in the output variables. By requiring the outputs to satisfy these virtual constraints, we re-instantiate the desired behavior. Intuitively, the output constraints can be added or removed so that the co-dimension of points of  $TQ_E$  that satisfy the active constraints is always  $\dim Q_E$ .

We have assumed  $\dim Q_E < \dim Q_R$ , and that the behavior specification  $(\phi, \omega, \eta)$  is satisfied by an example trajectory  $x_0$ , which was presumably obtained from a computationally intensive offline optimization, i.e. by the definition of the pullback

of a differential form:

$$\forall j, t : \omega_j(\varphi(x_0(t))) \cdot D\varphi(x_0(t)) \cdot x_0(t) = \eta_j(t, \varphi(x_0(t))) \quad (2.43)$$

From the constant rank assumption about  $\varphi(\cdot)$  we obtain that for each  $x_0(t)$  there is an entire manifold of possible values for a new instantiation  $x(t)$  such that  $\varphi(x(t)) = \varphi(x_0(t))$ . The original trajectory  $x_0$  clearly lies within this set, but should that trajectory become unrealizable due to damage, we can implement any of the other possible curves.

A different but equivalent perspective is the following. Assume the constraints  $\Omega_i$  are placed in order of priority; we indicate this priority by  $\gg$ :

$$\Omega_1 \gg \Omega_2 \gg \dots \gg \Omega_i \gg \dots \quad (2.44)$$

We are assuming that there are underlying physical constraints  $\Omega_P$ , design constraints  $\Omega_D$  derived from  $(\phi, \omega, \eta)$ , and learned constraints  $\Omega_L$ . The learned constraints are those that result from evaluating output constraints  $\omega$  on  $y_0(t)$ , thereby defining the appropriate  $\gamma_L$ . In this sense, the constraints are “learned”, rather than a priori known. These we combined into  $\Omega$  with  $\Omega_P \gg \Omega_D \gg \Omega_L$ , and are associated with  $\gamma_P$ ,  $\gamma_D$ , and  $\gamma_L$  respectively. We used the first  $\dim Q_R$  linearly independent constraints of these determine the velocity  $\dot{x}(t)$ . However, by virtue of the addition of  $\Omega_L$ , the number of constraints in  $(\Omega, \gamma)$  is larger than  $\dim Q_R$  (i.e.  $\Omega$  is “tall”), and these constraints are redundant on the instantiation of the behavior  $x_0(\cdot)$  - as we assumed maximum rank initially, adding rows cannot add rank. As long as the robot was functioning without damage, the  $\Omega_D(x_0) \cdot \dot{x}_0 = \gamma_D(t, x_0)$  constraints were satisfied by assumption, and no change in control was needed.

We modeled damage to to the robot as a low rank change to  $\Omega_P$ , replacing it with  $\tilde{\Omega}_P$  instead, of possibly lower or higher rank, but such that only a few constraints are

affected. In other words, only a few rows of  $\Omega_P$  and  $\gamma_P$  are changed due to damage. Consider the case where the rank change, i.e. change in number of constraints, associated with this damage to  $\Omega_P$  is such that

$$\dim Q_R - \text{rank}(\Omega)_L \leq \text{rank}(\tilde{\Omega}_P) + \text{rank}(\Omega)_D \leq \dim Q_R \quad (2.45)$$

When (2.45) holds, the change in rank induced by the damage can be taken up by removal or addition of learned constraints  $\Omega_L$ , and we could solve for new feasible velocities without harming compliance with any of the design constraints  $\Omega_D(x) \cdot \dot{x} = \gamma_D(t, x)$ .

Here the use of the dual representation in our behavior specification came into its own. It allowed us to gracefully recover from structural changes in the constraints governing the robot. If the explicit forms are known, finding a recovery trajectory required no optimization to be done – it follows directly from integrating the new behavior specification equation with the modified constraints; we demonstrated this in §2.5 below.

### 2.4.5 Canonical Cost Functions

In the case of a physical robot, the modified constraints might not be known; we explored this possibility by attempting to re-learn a walking behavior for a hexapedal robot. For this, optimization is a natural tool, as is the use of observation functions. We take an *observation function* to be a smooth map  $f : Q \rightarrow \mathbb{R}$ , where the domain  $Q$  will be made explicit. We consider them as “sensors” or “outputs” when evaluated along curves  $x(t) \subset Q$ . By our results from §2.4.3.2, we expect relatively arbitrary constraints to meet our requirements. For  $t \in [0, T]$ , control input  $u(t)$ , and trajectory  $x(t)$ , integral cost functions are extremely commonplace, i.e., A behavior specification

suggests the following cost function, for  $J[u]$  some choice of control cost :

$$\tilde{J}[x, u] := J[u] + \lambda \int_0^T \|\Omega(x) \cdot \dot{x} - \gamma(t, x)\|^2 dt \quad (2.46)$$

The cost is measuring the violation of the constraints, with weight  $\lambda \in \mathbb{R}$ . The constraints explicitly identify which directions of  $\dot{x}$  are relevant for tracking a desired output, versus which have no effects.

We used this approach in our hardware-in-the-loop optimization in section 2.6. In this task we were only concerned with the end-point of the robot motion and therefore we took the control cost  $J(u)$  to be 0. We have a manually-tuned gait which achieved our desired goal of moving forward in a straight line; since our technique provide an efficient *representation* of motion in the group, we expect to be able to reconstitute similar planar displacement via some gait. We then learned a behavior specification  $(\Omega_L, \gamma_L)$  using a choice of encoding template motivated by the Lateral Leg Spring ([Schmitt and Holmes, 2000](#)) and Spring Loaded Inverted Pendulum ([Blickhan, 1989b](#)) dynamic templates. We then ran the optimization it with the violation of constraints norm. This new optimization allowed our robot to re-learn the  $x_0$  trajectory within 36 attempts.

## 2.5 Crawler

We now present an example on a two-armed robot in simulation. The robot is depicted in Fig. 2.3. Each arm is a four-bar linkage that consists of four rigid bars connected end-to-end by powered swivel joints. Our objective will be to preserve the motion of the body when one of the joint actuators is jammed.

We take the configuration space  $Q_R$  of the robot to be  $(x, y, \theta, \theta_1, \dots, \theta_6) \in SE(2) \times \mathbb{T}^6 = G \times S$ . The joint angles are  $\theta := \{\theta_k\}_{k=1}^6 \in T^6$ , while the coordinates  $g = (x, y, \theta) \in SE(2)$  define the location and orientation of the body in the

plane. The initial condition of the robot's body is always  $(x(0), y(0), \theta(0)) = (0, 0, 0)$ , so that the  $(x, y, \theta)$  are simultaneously the coordinates for the body with respect to the standard world frame positioned at the origin.

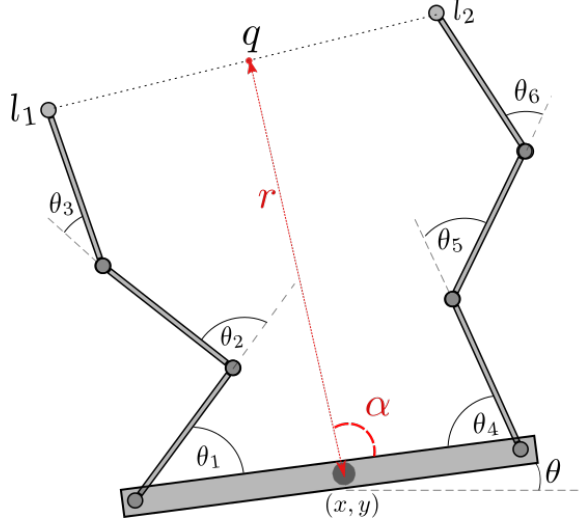


Figure 2.3: The grey members indicate components that belong to the robot. The points  $l_1$  and  $l_2$  are fixed foot locations. Each joint  $\theta_j$  is a powered rotational joint. The points  $h_1$  and  $h_2$  are the attachment points at which the limbs attach to the body. The center point of the foot locations  $q$  defines the value  $r$  and angle  $\alpha$  (in red), which are our choice of encoding template.

Our robot moved by attaching to the plane at the two locations  $l_1$  and  $l_2$  with freely rotating pivots, dragging itself with its limbs. Let us consider the problem of preserving this body motion when the two leg attachment points  $l_1$  and  $l_2$  are fixed, but a joint motor jams. While this example is artificially simple, it conveys our method. *Mutatis mutandis*, the extension to longer and more complex behaviors is analogous.

The robot is a kinematic system, so that using complex numbers to represent the plane, the following equations relate the points  $l_1$ , and  $l_2$  to the configuration variables  $\theta_i$ .

$$f_1(x, y, \theta, \theta_1, \dots, \theta_6) := l_1 = x + iy + e^{i\theta} \left( \sum_{i=1}^3 e^{i \sum_{j=1}^k \theta_j} \right) \quad (2.47)$$

$$f_2(x, t, \theta, \theta_1, \dots, \theta_6) := l_2 = x + iy + e^{i\theta} \left( \sum_{k=4}^6 e^{i \sum_{j=1}^k \theta_j} \right) \quad (2.48)$$

I.e., the robot is subject to the constraints  $f_1 - l_1 = 0$ , and  $f_2 - l_2 = 0$ , where functions  $f_i : Q_R \rightarrow \mathbb{C}$  are the R.H.S of Eqns. (2.47), (2.48). The equations (2.47), and (2.48) are equivalently holonomic constraints on the configuration space  $Q_R$  – they induce four real-valued constraint equations on the tangent bundle by differentiation. I.e.,

$$\begin{bmatrix} \nabla_{(g,\theta)}(\text{real}(f_1)) \\ \nabla_{(g,\theta)}(\text{imag}(f_1)) \\ \nabla_{(g,\theta)}(\text{real}(f_2)) \\ \nabla_{(g,\theta)}(\text{imag}(f_2)) \end{bmatrix} \begin{bmatrix} \dot{g} \\ \dot{\theta} \end{bmatrix} = \begin{bmatrix} 0 \\ 0 \\ 0 \\ 0 \end{bmatrix} \quad (2.49)$$

We write (2.49) compactly as

$$\omega_A(g, \theta) \cdot (\dot{g}, \dot{\theta}) = 0 \in \mathbb{R}^4. \quad (2.50)$$

$\omega_A \in (T^*(T^6 \times SE(2)))^4 \cong \mathbb{R}^{4 \times 9}$  is the Jacobian matrix which multiplies  $(\dot{g}, \dot{\theta}) \in T(T^6 \times SE(2)) \cong \mathbb{R}^9$ . In the  $(g, \theta)$  coordinates, the Jacobian appears in block form as:

$$\begin{bmatrix} \omega_A^g & \omega_A^\theta \end{bmatrix} \begin{bmatrix} \dot{g} \\ \dot{\theta} \end{bmatrix} = 0 \in \mathbb{R}^4, \quad \omega_A^g \in \mathbb{R}^{4 \times 3}, \omega_A^\theta \in \mathbb{R}^{4 \times 6} \quad (2.51)$$

In the notation of §2.4, Eqn. (2.49) is  $\Omega_P$ , as they are mandatory constraints that enforce the physics of the crawler, i.e.,

$$\Omega_P^g \cdot \dot{g} + \sum_{i=1}^6 \Omega_P^{\theta_i} \cdot \dot{\theta}_i = \gamma_P(t, x, y, \theta, \theta_1, \dots, \theta_6) = 0 \quad (2.52)$$

The choices of parameters  $l_1, l_2, h_1$ , and  $h_2$  are shown in Table. 2.2.

Table 2.2: Parameters

Parameter	Value
$l_1$	$+2.5 + i2$
$l_2$	$-2.5 + i2$
$h_1$	$+1$
$h_2$	$-1$

We now define a choice of encoding template  $Q_E$ . Define the point  $q \in \mathbb{C}$  to be the midpoint of the two feet:

$$q = \frac{l_1 + l_2}{2}$$

We take  $(r, \alpha) \in \mathbb{R} \times S^1$ , and define our template system with the point  $q$  via Eqn. (2.53).

$$r \exp(i(\theta_0 + \alpha)) + x + iy = q \quad (2.53)$$

Eqn. (2.53) defines an implicit relation between the template shape variables  $(r, \alpha)$  and the anchor shape variables  $\theta$ .

Thus we have the encoding template  $Q_E = SE(2) \times \mathbb{R} \times S^1$  as the body frame, with additional  $(r, \alpha) \in \mathbb{R} \times S^1$ , as shown in red in Fig. 2.3. The template shape coordinates are the distance  $r$  of the CoM to the point  $q$ , while  $\alpha$  is the angle of the CoM with respect to the line between  $q$  and the CoM.

Explicitly, we can compute  $r$  and  $\alpha$  as follows. Define the two auxiliary functions  $p_1 : T^6 \rightarrow \mathbb{C}$ , and  $p_2 : T^6 \rightarrow \mathbb{C}$  as:

$$\begin{aligned} p_1(\theta) &:= h_1 + \exp(i\theta_1)(1 + \exp(i\theta_2)(1 + \exp(i\theta_3))) \\ p_2(\theta) &:= h_2 + \exp(i\theta_4)(1 + \exp(i\theta_5)(1 + \exp(i\theta_6))) \end{aligned}$$

By solving (2.53) for  $(r, \alpha)$  in terms of  $\theta$ , it can be shown that the phase map  $\varphi^s :$



$T^6 \rightarrow (r, \alpha) \in \mathbb{R} \times S^1$  is given Eqn. (2.54).

$$\varphi^s(\theta) = \left( \left\| \frac{(p_1 + p_2)}{2} \right\|, \angle \frac{(p_1 + p_2)}{2} \right) \quad (2.54)$$

Thus, our total map phase map is  $\varphi = (\varphi^s(\theta), \text{id}_G)$ . Equivalently, the variables  $(x, y, \theta, r, \alpha)$  define the five outputs  $(y_1, \dots, y_5)$ . Our desired template reference trajectory is shown in Fig. 2.4. The resulting group velocity is shown in Fig. 2.6 under the “desired” label. We denote this reference template signal as  $y(t) = (g_d(t), r_d(t), \alpha_d(t))$ ,  $t \in [0, 1]$ . An anchor trajectory that produces the desired phase reference is shown in Fig. 2.5. The anchor angles used to generate this curve are denoted by  $\theta_{org}(t) \in T^6$ ,  $t \in [0, 1]$ . In the notation of §2.4.4,  $x_0(t) = (g_d(t), \theta_{org}(t))$ .

Let us now construct the complete behavior specification  $(\varphi, \omega_D, \gamma_D)$ . The template coordinates obey two Pfaffian constraints determined via Eqn. (2.53), which we write as.

$$\omega_T(g, r, \alpha) \cdot (\dot{g}, \dot{r}, \dot{\alpha}) = 0 \in \mathbb{R}^2 \quad (2.55)$$

We denote the associated block structure via  $\omega_T^g \in \mathbb{R}^{2 \times 3}$ ,  $\omega_T^s \in \mathbb{R}^{2 \times 2}$ , analogous to Eqn. (2.51). These template constraints are constraints on  $Q_E$  (equivalently, the output variables); so,  $\varphi^* \omega_T$  are members of  $\Omega_D$ .

If the block  $\omega_T^g$  were invertible, we could write a world-coordinate version of the reconstruction equation shown in Eqn. (2.11) as  $\dot{g} = -(\omega_T^g)^{-1} \omega_T^\theta$ . We note that we have  $\dot{g} \in T_g SE(2)$ , as opposed to  $\xi \in \mathfrak{se}(2)$  as literally shown in Eqn. (2.11). While we could transform  $\dot{g}$  to the Lie algebra via  $T_g L_{g^{-1}}$ , our expression is sufficient for our example as-is. We are only considering one “stance-phase” where world coordinates are adequate.<sup>6</sup> Analogously, we could write a reconstruction equation on the template if  $\omega_A^g$  was invertible.

---

<sup>6</sup>Furthermore, it illustrates that our approach of virtual constraints on template models is more general than the kinematic lifting shown in §2.4.3.2. The construction presented there is *sufficient* and *explicit*, but not the only case where virtual constraints would work., e.g., without group invariance.

We emphasize that a psuedoinverse of  $\omega_T^G$  is not a satisfactory substitute for a true inverse. Our *key* requirement, as shown earlier in Eqn. (2.18), is  $\omega_g^T$  to be full rank on  $Q_E$ , so a third independent constraint is necessary. Equivalently, for a given shape value  $(r, \alpha) = \varphi^s(\theta)$ , there must be a unique  $\dot{g}$  such that Eqn. (2.51) holds. Ergo, we need to synthesize another constraint from an observation function  $f : Q \rightarrow \mathbb{R}$ , as in §2.4.3.2. Recall, our argument that we are able to do this is based upon recovering a high codimension object – namely, a trajectory on the template. As we saw in §2.4.3.2, there is an open and dense set of functions that are suitable. Since we are designing a behavior specification, we choose the constraint (which is satisfied by our desired group motion)

$$\dot{x} = \dot{\theta} \quad (2.56)$$

This gives  $\omega_3 := \mathbf{d}x - \mathbf{d}\theta \in T^*Q_E$ , and  $\gamma_3(t, r, \alpha, x, y, \theta) := 0$ .

We augment Eqn. (2.55) with the row defined by Eqn. (2.56) exactly as shown in Eqn. (2.32) of §2.4.3.2 to get augmented form  $\tilde{\omega}_T$  so that the sub-block  $\tilde{\omega}_T^g \in GL(3, \mathbb{R})^7$ . The two rows of  $\omega_T$  are the constraints on  $Q_E$  that implicitly relate the  $g$  coordinates to the  $(r, \alpha)$  coordinates - along with our design constraint  $\omega_3 := \mathbf{d}x - \mathbf{d}\theta$ ,  $\gamma_3 := 0$ , we have three constraints represented by  $\tilde{\omega}_T$ . Since  $Q_E$  is five dimensional, we must produce two additional constraints. The next two constraints are simply the condition Eqn. (2.16) in the  $(r, \alpha)$  coordinates. In  $(r, \alpha)$  coordinates, the constraint and value pairs are  $\omega_1 := \mathbf{d}r$ ,  $\gamma_1 := \frac{d}{dt}r_d(t)$ ,  $\omega_2 := \mathbf{d}\alpha$ ,  $\gamma_2 := \frac{d}{dt}\alpha_d(t)$ ,

We thus have a complete specification  $(\varphi, \omega, \gamma_D)$ , for  $g = (x, y, \theta)$

$$\varphi^* \tilde{\omega}_T^g \cdot \dot{g} + \varphi^* \tilde{\omega}_T^{(r, \alpha)} \cdot (\dot{r}, \dot{\alpha}) = 0 \quad (2.57)$$

$$\mathbf{d}r \circ (\dot{r}, \dot{\alpha}) = \frac{d}{dt}r_d(t) \quad (2.58)$$

$$\mathbf{d}\alpha \circ (\dot{r}, \dot{\alpha}) = \frac{d}{dt}\alpha_d(t) \quad (2.59)$$

---

<sup>7</sup>We numerically verified that rank is three in a neighborhood of the goal trajectory

The L.H.S is  $\omega$ , the R.H.S is  $\gamma_D$ , and  $\varphi$  is obvious. We emphasize that the behavior specification exists independently of the physical constraints  $\Omega_P$ , and are *satisfied by definition* when the robot executes  $x_0(t)$ ; the behavior specification was not used to define  $x_0$ , but is a consequence of it.

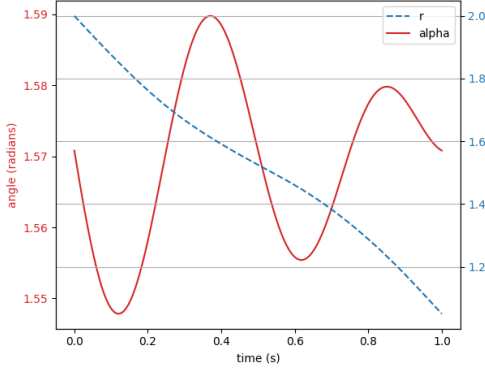


Figure 2.4:  $r$  and  $\alpha$  coordinates of the template along the  $\theta_{org}$  curve, i.e. - the desired reference trajectory on the template.

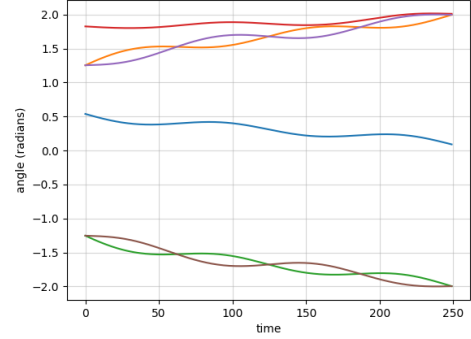


Figure 2.5: The six joint angles of  $\theta_{org}$

It might seem strange to the reader that we have elected to split the configuration variables into group and non-group components, even though our reference trajectory  $y(t)$  is in both coordinates; i.e. we could potentially fit  $\omega$  directly, without separating the components. Our motivation for doing this is two-fold. First, the weaker reason, is to tie it to the perspective of §2.4.3, and two, more strongly, we wish to solve the constraints for an *input*  $(\theta_1, \dots, \theta_6)$ . The group variables  $g$  are not directly controlled. We will see in the sequel that this “connection-like” separation allows us to isolate the joint angles so that we can solve for the  $\theta_i$  in terms of  $g$  and the constraints. We will see this allows a particularly nice representation.

We now assume that the  $\theta_1$  actuator is jammed, so that  $\forall t, \theta_1(t) = \theta_1(0)$ . This is constant for all gaits, as we have also assumed the initial condition is fixed. If we did nothing, and simply played back the same  $\theta_{org}$  with the  $\theta_1$  actuator stuck, we obtain considerable error. In Fig. 2.6 the curve labeled “old” illustrates the resulting

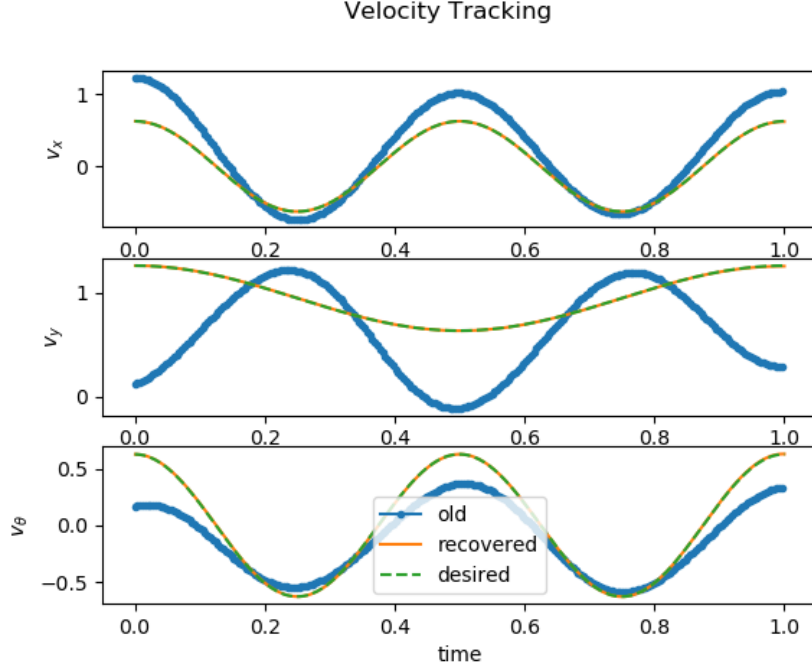


Figure 2.6: The CoM group velocity  $\dot{g} = (v_x, v_y, v_\theta)$  over time. The “desired” curve is the undamaged motion that we are aiming to recover. The blue “old” curve is the group velocity achieved if no recover strategy is attempted post damaged. The “recovered” trace is the performance after recovery.

group velocity of the CoM should “playback” be attempted without recovery. Ergo, attempting to improve this via exploiting the redundancy of the limbs is warranted.

We pull-back  $\omega_1, \omega_2$ , and  $\omega_4$ , to  $Q_R$  to define  $\Omega_D$ , and with the jamming constraint, we obtain a six-dimensional non-autonomous differential equation that can be integrated in forward time. The solution of this equation have components  $\theta_i$  that generate the same desired COM motion despite the seized limb, if such a solution exists. However, the solutions to this equation need to *also* satisfy the  $\Omega_P$  constraints. As we have the model equations (2.57) and (2.52), we can write the explicit system of differential equations to encode our recovery strategy subject to this requirement.

We first substitute  $y$  in Eqn. (2.57),

$$\dot{g}_r(t) = (\tilde{\omega}_T^g)^{-1} \tilde{\omega}_T^{(r,\alpha)} \cdot (r_d(t), \alpha_d(t)) \quad (2.60)$$

This is an autonomous differential equation that we know our curve  $y$  satisfies. Let  $\theta_V = (\theta_1, \dots, \theta_6)$ . Since we are seeking a  $\theta_V(t) \neq \theta_{org}, t \in (0, 1]$  that satisfies Eqn. (2.17), we substitute Eqn. (2.60) into Eqn. (2.52) to get

$$-\left(\omega_A^g \cdot (\tilde{\omega}_T^g)^{-1} \tilde{\omega}_T^{(r,\alpha)} \cdot (r_d(t), \dot{\alpha}_d(t))\right) = \omega_A^{\theta_V} \cdot \dot{\theta}_V \quad (2.61)$$

Observe that Eqn. (2.61) is also a non-autonomous differential equation, as the coefficients  $\omega_A^g$  and  $\omega_A^\theta$  depend on  $\theta_V$ . We additionally require that Eqn. (2.16) is satisfied, so we write

$$\varphi^s(\theta_V(t)) = (r_d(t), \alpha_d(t)) \implies D_{\theta_V(t)} \varphi^s \cdot \dot{\theta}_V(t) = (r_d(t), \dot{\alpha}_d(t))$$

We employ the Moore-Penrose pseudoinverse to write this as the non-autonomous differential equation,

$$\dot{\theta}_V(t) = D_{\theta_V(t)}^\dagger \varphi \cdot (r_d(t), \dot{\alpha}_d(t)) \quad (2.62)$$

Using the approach in Eqn. (2.62) and (2.61), and the constraint that  $e_1 \cdot \theta_V = 0$ , we write a combined system as the following

$$\dot{\theta}_V = \begin{bmatrix} \omega_A^{\theta_V} \\ D_{\theta_V(t)} \varphi \\ e_1 \end{bmatrix}^\dagger \begin{bmatrix} -\left(\omega_A^g \cdot (\tilde{\omega}_T^g)^{-1} \tilde{\omega}_T^{(r,\alpha)} \cdot (r_d(t), \dot{\alpha}_d(t))\right) \\ (r_d(t), \dot{\alpha}_d(t)) \\ 0 \end{bmatrix} \quad (2.63)$$

The inclusion of the jamming constraint is the modification of  $\Omega_P$  to  $\tilde{\Omega}_P$  we described in §2.4.4. In the  $\Omega$  notation, if  $\Omega_s$  has an inverse  $\Omega_s^\dagger$ , we can obtain for any given  $s(t)$ :

$$\dot{s} = \Omega_s^\dagger(g, s) (\gamma(t, g, s) - \Omega_g(g, s) \cdot \dot{g}), \quad (2.64)$$

This equation is the “dual” of the reconstruction equation, in that we are solving for

shape in terms of group. Since we are assuming that  $s$  has sufficient redundancy to re-implement the behavior, the matrix  $\Omega_s$  is “wide”, so is not invertible in our typical use case.

Eqn. (2.63) is a non-autonomous, differential equation. We implemented it numerically in Python 2.7.5 with the `numpy` and `scipy` numerical processing libraries. We cannot naively numerically integrate Eqn. (2.63). We require that the constraints are enforced. Eqns. (2.57) and (2.52) must hold *exactly*. Conventional integrators are approximating the true solution, so the error compounds as the integration time grows. If the constraints are violated too badly, it is not reasonable to consider the resulting trajectory segment a meaningful solution.

We used the finite-step-size method `lsoda` from the `scipy.integrate.ode` package. Provided with  $x_i, t_i$  and vector field  $f$ , it returns  $x_{i+1}$ , and is thus called in a loop to produce a sequence  $\{t_i, x_i\}_{i=0}^{t_n}$ . At each  $x_i, t_i$ , we projected  $x_i$  onto the constraint manifold using Newton iteration to point  $p_i$ , then passed  $p_i, x_i$  into  $f$ , i.e.

```

ti = 0
xi = x0
while ti < t1:
    ti, xi = integrate(ti, xi, t+dt)
    ti, xi = projectConstraints(ti, xi)

```

Employing this crude form of geometric integration on Eqn. (2.63), we obtained a new  $\theta_{rec}(t)$ ,  $r(t)$ , and  $\alpha(t)$ , shown in Figs. 2.4 and 2.7. Note that we cite Fig. 2.4 for the new phase curve as well. This is intentional, as the new and old phase curves are numerically identical.

The performance of our recovered joint inputs is shown in Fig. 2.6 as the “recovered” trace. As we can see, it appears to recover the desired group velocity (and thus, group position) quite well, especially in relative performance to no recovery at all.

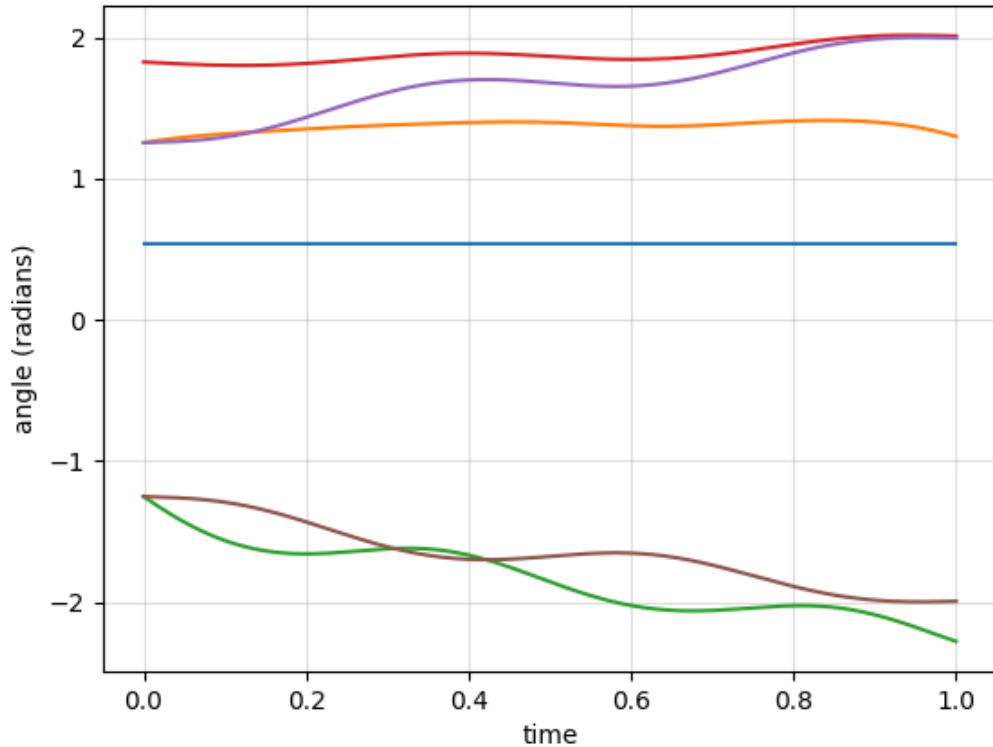


Figure 2.7: Recovered joint angle inputs. The  $\theta_1$  (blue) trace is visibly seen to be constant, reflecting the jammed state.

## 2.6 Enepod

We built a six-legged robot to determine if our strategy is achievable on a actual device.

To test this approach on a physical device, we looked for a justifiable choice of encoding template. For biomechanists, the difference between “running” and “walking” is defined in terms of the energy reservoirs participating in the exchange generating the motion. In walking, potential energy exchanges with kinetic energy by vaulting over a rigid leg; thus ground speed is lowest when the center-of-mass is highest. In running, elastic energy of stretched tendons and muscles exchanges with kinetic energy; thus ground speed is highest when the center-of-mass is highest. So, we designed a six-legged robot to facilitate the measurement of elastic energy storage in its legs.

This, we hoped, would allow us to define an encoding template in terms of these energy exchanges, and determine if our strategy is achievable on a physical device.

The robot (which we refer to as the “enepod”) is depicted in Fig. 2.8. It consists of a 7DoF sequence of Robotis modules (Robotis Dynamixel EX and MX) as an actuated backbone, while the six leg are constructed of  $\frac{1}{16}$ -inch 1075 spring steel, which is flexible enough that it generates deflections of greater than 1 cm at the foot during motion. We made the legs compliant enough so that their deflections would be easy to sense using a motion tracking system, providing an instantaneous window into the elastic energy stored in the body at any given time.

We generated the robot gait by feedforward time-sequenced position commands which are carried out by individual PID control loops in the motor modules. The gait we chose is an “alternating tripod” gait analogous to that used in the RHex hexapod (*Saranli et al., 2001*). The feet are grouped into two collections of three that are in phase (a “tripod”), while the other three are  $\pi$  out-of-phase. If the system were perfectly rigid, each tripod would be undergoing an identical motion. With springy legs, while the legs receive the same commands, the dynamics of the legs springs alter the response.

The aim of experiment is to perform a hardware-in-the-loop optimization to construct an input that achieves effective forward motion where the cost function is determined via our constraints. Our hypothesis is that the robot will learn to move forward quite rapidly using the behavior specification.

As opposed to physically damaging the robot, we have a parametric space of gaits  $\mu \in [-1, 1]^4$ . We are going to demonstrate that the output constraints are sufficient to generate a gait closer to the desired motion. Since we have zero model information for the enepod, we argue this analogous to recovering from damage, where the model is also unknown<sup>8</sup>. The values of  $\mu$  determine the signal that drives the center module

---

<sup>8</sup>Future work will compare pre- and post-damage performance on a single platform



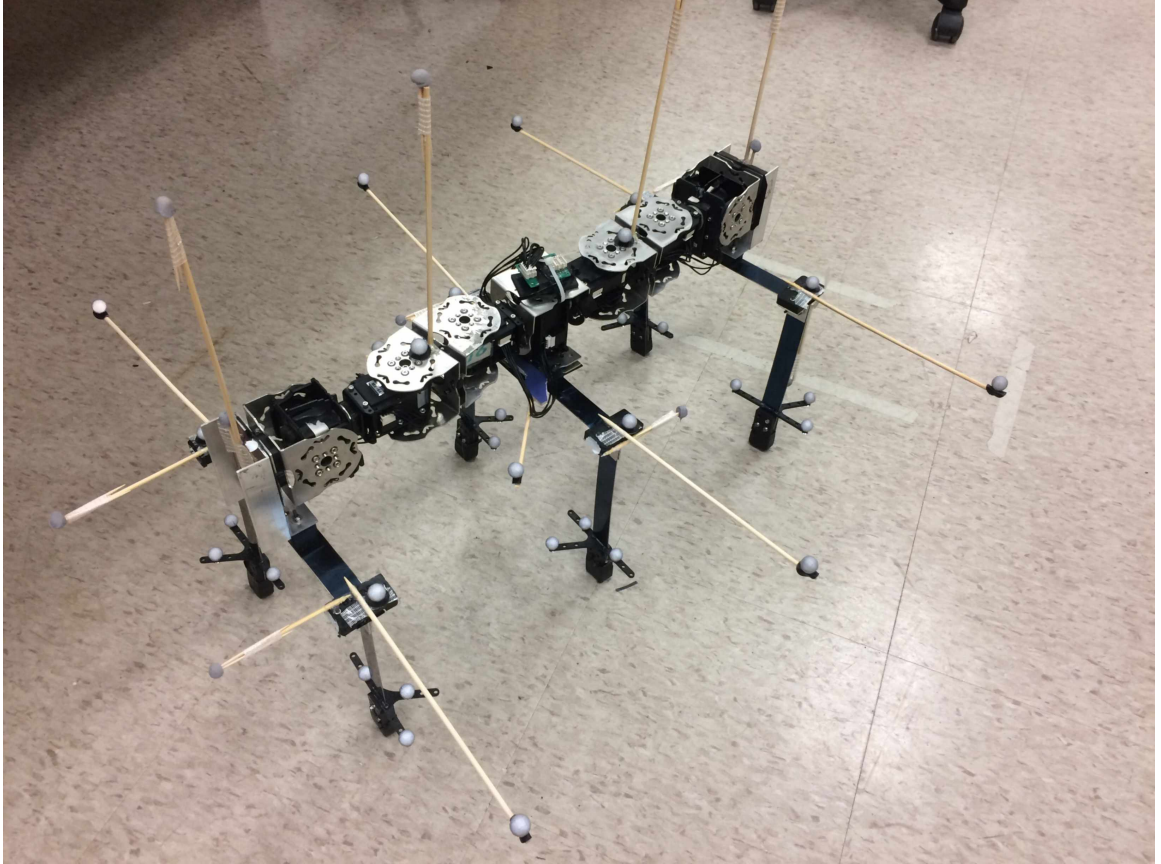


Figure 2.8: The enepod robot. The wooden sticks mount retroreflective markers that are used by a motion capture system to capture configuration data in real time. The motion capture system is part of the hardware-in-the-loop optimization setup used to generate motion.

$MC$  (see Fig. 2.9). The other six modules' input signal remains unchanged. Since the number of constraint functions exceeds the dimension of the group, which is taken to be  $SE(2)$  (symmetry in the plane), we expect that the group motion is completely defined (and indeed will be, if the rank conditions are satisfied).

To expand on this, the encoding template we will define is motivated by the Lateral Leg Spring ([Schmitt and Holmes, 2000](#)) and SLIP dynamic templates. We will sum the spring deflection of each leg per tripod as a approximate of the elastic energy stored in each member, as energy stored scales monotonically with deflection for conventional spring models.

We take that our projection  $\varphi$  aggregates tripod variables but otherwise leaves

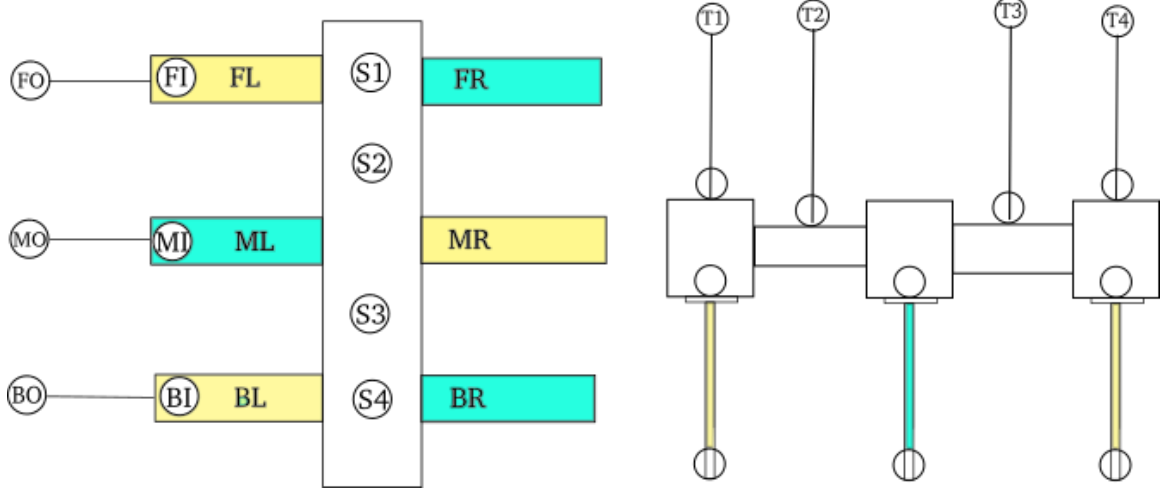


Figure 2.9: Leg groupings on the Enepod- the (BL,MR,FL) legs maintain phase, while the (BR,ML,FL) are collectively out of phase with the first tripod.

the group member fixed, as above, i.e.,

$$\varphi(s, g) = (\varphi^s, g) \quad (2.65)$$

Then, motivated by ([Schmitt and Holmes, 2000](#)), we assume that grouping the legs into tripods obeys, for elastic energy function  $E$  (for notational simplicity, we assume there is one function  $E$ , but strictly speaking each spring has its own function  $E_l$ )

$$E(\sum_{i \in I} \varphi_i) = \sum_{i \in I} E(\varphi_i) \quad (2.66)$$

where  $I$  is an index set for a given leg. I.e. that the energy of the springs is *additive*; while energy is an additive quantity generally, we are emphasizing that our map  $\varphi$  aggregates the energy per-tripod.

### 2.6.1 Methods

Our experimental setup consists of three major components – the robot itself, the motion tracking system, and the optimization software.

The robot, as described above, is a collection of Robotis modules - the fore, aft, and center modules, which connect to the legs, are EX modules, while the remaining are MX modules. It is not autonomous in any way – each module tracks a reference signal provided from an external source. The motion tracking system is the commercially available Qualisys tracking system using 10 Opus-310 cameras collecting data at 250 FPS, managed through the Qualisys Track Manager software, version 2.17. In the capture arena, the system provides time-series position measurements of the retroreflective markers. The Qualisys telemetry is collected by the optimization software package, which is responsible for both filtering the telemetry, updating the gait parameters, and driving the robot.

Our optimization problem occurs in two stages. In the first stage, we manually designed a known gait that was further optimized with a “conventional” goal function that evaluates effective displacement per cycle. In other words, we intelligently seeded the optimizer with a working gait that was further refined via optimization. The resulting gait is the “ $x_0(t)$ ”, from above. We then evaluated our chosen observation functions along this cycle, and constructed a representation of the functions with a Fourier series fit; these are functions defined on  $Q_E$  as functions of phase. Using this model, we differentiated to obtain the necessary  $\omega_i := dy_i$ , and  $\gamma_i$ . These resulting functions of phase are the “learned constraints”  $\Omega_L$  that we introduced in §2.4.4. These  $\Omega_L$  will define the cost function  $J$ , exactly as in Eqn. (2.46).

In the second stage, we then re-initialized the robot with gait parameters  $\mu_1 \in [-1, 1]^4$  that were *not* a viable gait. In this case, they caused the robot to have an average displacement of zero. The goal of the optimization is to generate a new set of feasible gait parameters. Unlike the preceding cases, the parameter space is not restricted in any way. In this setting, we would expect that a successful optimization can recover the a forward motion. While the theory predicts exact recovery, the limitations of practical hardware suggests improvement of a gait, rather than the

complete regeneration of one.

### 2.6.1.1 Coordinates and Instrumentation

Our motion tracking data consists of labeled time sequences of position data that gives  $(x, y, z)$  coordinates of each marker at a time sequence  $\{t_i\}$ . The Qualisys tracking data was processed by a custom Python 2.7.5 script that filtered and scored the data. Each marker has a unique name for which we will consistently reference. The marker names are depicted in Fig. 2.9. A marker  $A$  has  $(A_x, A_y, A_z) \in \mathbb{R}^3$  spatial coordinates with respect to an arbitrary world frame that arises by calibrating the motion tracking system.

Since we by definition are interested in the displacement of the CoM, we treat the robot as a rigid body with the legs attached. However, the spine is articulated, so we define a virtual rigid body  $C$  as a proxy using the markers S1, S4, VR, and VL. We denote the CoM of  $C$  as  $(C_x, C_y, C_z) \in \mathbb{R}^3$ . As we are using  $C$  merely as a location, we assume each marker is unit mass, so that the CoM is just the mean of the S1, S4, VR, and VL locations.

The robot has coordinates for each leg, as shown in Fig. 2.9. We will also numerically refer to an entire leg by an index, as shown in the figure. E.g, Leg 2 refers to the entire assembly of the leg  $ML$  and associated markers attached to it.

### 2.6.1.2 Gait Space

While the module we control has five gait parameters  $\mu_1$ , the remaining motors need inputs as well. There are seven Robotis modules, so we need to define seven input signals to fully realize a gait. We represent this space with 28 parameters, four for each channel. Each of the four parameters defines a knot point of a linearly interpolated signal. The  $x$ -coordinate of each knot point is fixed at  $(0, 0.25, 0.5, 0.75, 1)$ , respectively. The last knot point at  $x = 1$  is set in software to equal the first,

so that the resulting signal is periodic with  $T = 1^9$ . The parameters then define the  $y$ -coordinates, allowing the representation of various “triangle” waves, as depicted in Fig. 2.10. We chose this parameterization as our initial reference gait was composed of sinusoids. The mechanical bandwidth of the robot is such that the motion between re-sampling a sine wave with our parameter scheme, versus using a sine function is software, was indistinguishable.

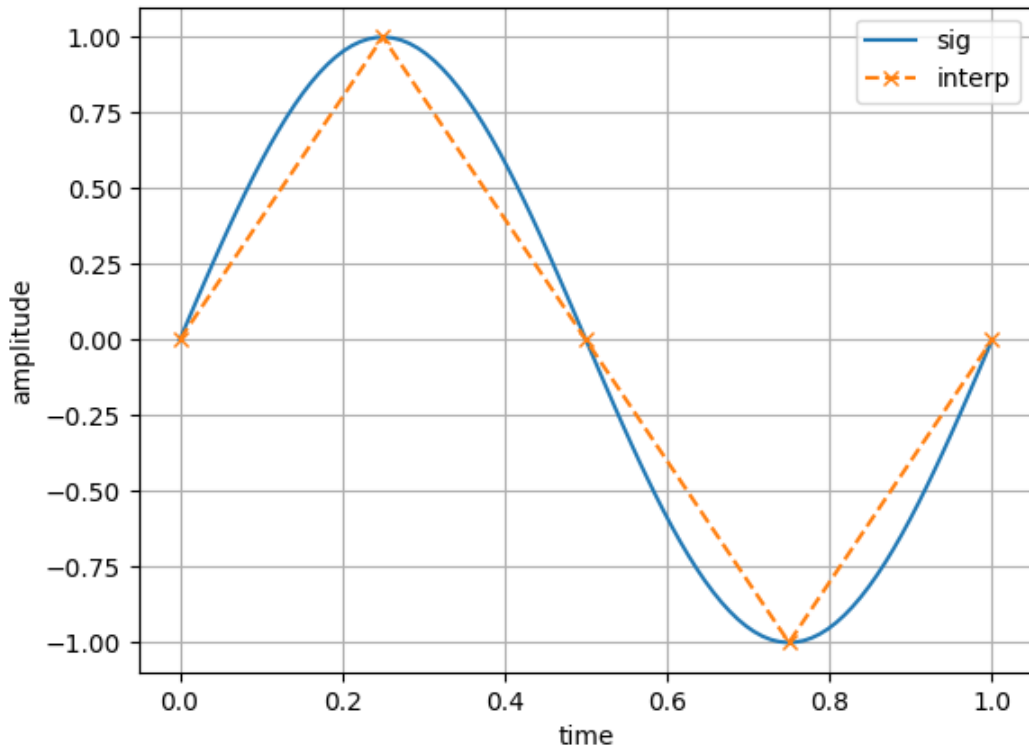


Figure 2.10: Our parametric scheme for representing input signals. The “sig” trace is a desired nonlinear function, while “interp” is our linear interpolation of it. Our parameterization adjusts the amplitude of the knots, but not their horizontal location. The sawtooth approximation of a sine wave is operationally equivalent on the robot.

---

<sup>9</sup>The software rescales it to the desired frequency when driving the robot

### 2.6.1.3 Phase

A priori, the dynamic phase of a limit cycle depends on dynamic information. However, we operationally assumed that there was a “kinematic phase” (Revzen *et al.*, 2009) that was defined only in terms of the kinematic variables. Near a periodic orbit, the behavior is one-dimensional, so that the kinematic state and dynamic state are bijectively related (otherwise, it would not be periodic); motivated by this, we estimate kinematic phase and use it as a local proxy for the true phase. Ergo, our phase estimate was produced using only position and time measurements.

The phase map  $\varphi$  is estimated using the numerical tool *Phaser* (Revzen and Guckenheimer, 2008) to estimate a phase map. We operationally assumed that the tool will produce a workable phase map, ignoring any theoretical assumptions required (i.e. - we applied the tool to the measured data, and found the resulting function satisfactory). Recall, we only require *any* retraction, rather than proper *asymptotic phase*.

The utility *Phaser* expects that the limit cycle of interest is contained within a two dimensional plane. To satisfy this requirement, and to ensure that our phase map is a map from *configuration* space, we project onto the first two principal components of the nominal gait position data, then ran *Phaser* on the resulting trajectories. The resulting phase map was cached for use in all evaluations of the cost function.

As this principal component projection occurs at every evaluation of the cost function, we chose an initial basis for this space when the nominal gait was executed. We then cached this choice of basis for subsequent projections, insisting that the subspace was well-defined and consistently oriented for all trials.

### 2.6.1.4 Observation Functions

There are four observation functions used. The vertical deflection of each tripod, and the horizontal deflection of each tripod. We assume the legs have no meaningful

mass in comparison to the spine, so that they contribute no momentum as masses in their own right. The legs only generate forces on the body through their deflection. We furthermore assume that the springs are perfectly one-dimensional – they are only able to deflect in their principal (softest) direction. Since stiffness of beams scales quadratically with thickness, and inverse cubically with length, the legs are significantly softer in the longest direction. We separated horizontal and vertical spring deflections, as by design, these deflections were dominated by different collections of springs.

An intuition behind this choice of output is that each of the tripod’s constituent legs move together on the nominal gait; ergo, we expected aggregating the six legs into two tripods to be an informative reduction. As we saw in §2.4.3.2, such intuition is not strictly necessary – a large number of arbitrary choices would theoretically be acceptable. However, we expect that not all choices are equally well-conditioned. While the resulting  $\Omega$  may have the desired rank in the ideal case, the condition number could be very poor in practice. Our argument from stiffness scaling, combined with empirical verification, suggests that the tripod deflections that we have selected for outputs are quite independent of each other, hopefully alleviating this concern.

We record a nominal value for the observation function by measuring its value while the robot is executing the goal behavior. By computing and caching Fourier-series approximations of the observation functions along the desired trajectory, we develop the constraints as we did in §2.4.3.2. As we will see, the deflections are strongly periodic, so a low order fit of  $n = 4$  for the Fourier series was taken as satisfactory. The quality-of-fit of the constraints for the disturbed robot will be appear as the cost-function in §2.6.1.7. We will refer to the signals defined by the vertical and horizontal spring deflections along the nominal gait a  $V_{ref}(\phi) \in \mathbb{R}^2, \phi \in [0, 2\pi]$ , and  $H_{ref}(\phi) \in \mathbb{R}^2, \phi \in [0, 2\pi]$ , respectively. When we evaluate these functions along the  $i$ -th iterate, we will use the notation  $V_i$  and  $H_i$ , respectively.

### 2.6.1.5 Vertical Spring Deflection

The vertical spring deflection  $V_i$  is calculated from telemetry by taking the arc cosine of the angle determined by the inner product  $\langle SN - TN, nO - nI \rangle$ , where  $n$  denotes the leg index, as shown in Fig 2.9. I.e,

$$\begin{aligned} V_1 := & \arccos(\langle S1 - T1, FLO - FLI \rangle) \\ & + \arccos(\langle S2 - T2, MRO - MRI \rangle) \\ & + \arccos(\langle S4 - T4, BLO - FLI \rangle) \end{aligned} \quad (2.67)$$

$$\begin{aligned} V_2 := & \arccos(\langle S1 - T1, FRO - FRI \rangle) \\ & + \arccos(\langle S2 - T2, MLO - MLI \rangle) \\ & + \arccos(\langle S4 - T4, BRO - FRI \rangle) \end{aligned} \quad (2.68)$$

We have assumed that the angle is sided (e.g., there are six distinct angles, as we are ignoring any geometric constraints across the body), despite the left and right angles resulting from the deflection of the same piece of spring steel, as the crossbeam is clamped to the body in the middle. Fig 2.15 depicts an example time series collected from the enepod striding forward with its limbs cycling at 2.5 Hz. While the x-axis is given in the number of sampled points rather than time, it was communicated with TCP and resampled, so that there are no missing packets and a constant  $\Delta t$ . The three deflection signals were added together for each tripod, reducing the six signals to two. We expect this to be a reasonable aggregate, as the gait alternates which tripod is contacting the ground. It also suggests the redundancy we assumed the robot has initially exists; we are not interested in individual *forces*, but rather we would like the total wrench induced by the legs on the spine to remain invariant. Regardless of the number of legs employed, if the system remains controllable in the



necessary directions, it would be possible to do so.

Our computation procedure is as follows. We first use the rigid body (as defined in §2.6.1.1) to transform the marker data into the body frame. E.g, see Fig. 2.12. In these coordinates, the body motion is periodic. Our procedure is to slice an execution of the robot into strides, via the phase map computed in §2.6.1.3. Computing the functions in Eqn. (2.67), and Eqn. (2.68) on the nominal gait, we obtain the signals depicted in Fig. 2.16.

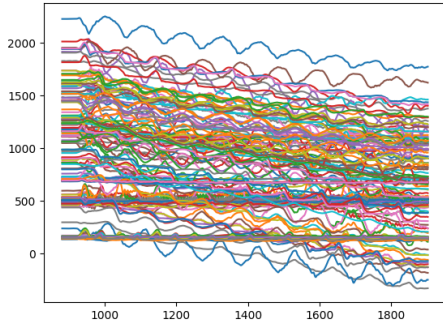


Figure 2.11: The filtered marker traces in world coordinates for an execution. The robot starts stationary, walks for a time, and then stops.

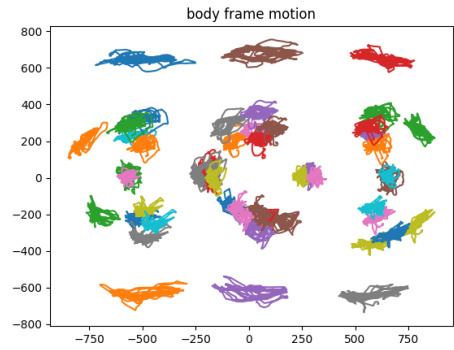


Figure 2.12: The same traces transformed into body coordinates using the virtual rigid mass. In these coordinates, the bias due to translation in the world frame is removed.

There apparently seems to be two distinct periodic signals approximately  $\pi$  out-of-phase with respect to each other, irrespective of their relative magnitudes. The reason for the bias in magnitude from one tripod to another is unknown.

While the structure of the summed effect is apparent, when re-visualized as a function of phase, an even more predictable signal emerges, shown in Fig. 2.13. The tripod's vertical deflection seems to be functionally related to the phase estimate, i.e. quite periodic.

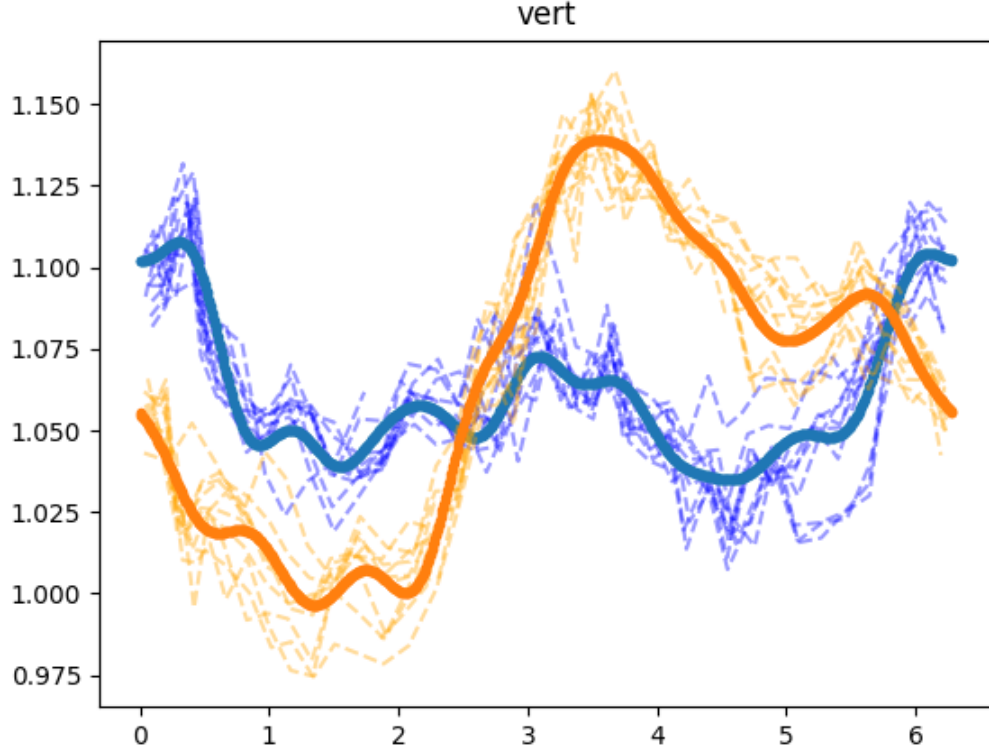


Figure 2.13: The summed vertical deflection signals for tripod one and two. The dashes lines are over a single stride, while the solid lines are Fourier-series ( $n=5$ ) fits of the measured data.

#### 2.6.1.6 Horizontal Spring Deflection

The fore-aft deflection of the spring vertically-mounted springs (e.g., the deflection in the “horizontal” direction) was measured differently from the vertical deflection. The marker sets indicated in Fig. 2.8 were used to define the two centroids,  $C_{top}^i$ , and  $C_{foot}^i$ , respectively, where  $i$  indexes the leg. We define the signal, where  $\pi_{xy} : \mathbb{R}^3 \rightarrow \mathbb{R}^2$  is projection onto the  $xy$ -plane, by

$$fa^i(t) := \pi_{xy} (C_{top}^i - C_{foot}^i) \quad (2.69)$$

If we merely repeated the aggregating we did in Eqn. (2.67) and Eqn. (2.68), we did not see a meaningful signal. In Fig. 2.17, we plot heatmaps of the  $fa^i(t)$ . The

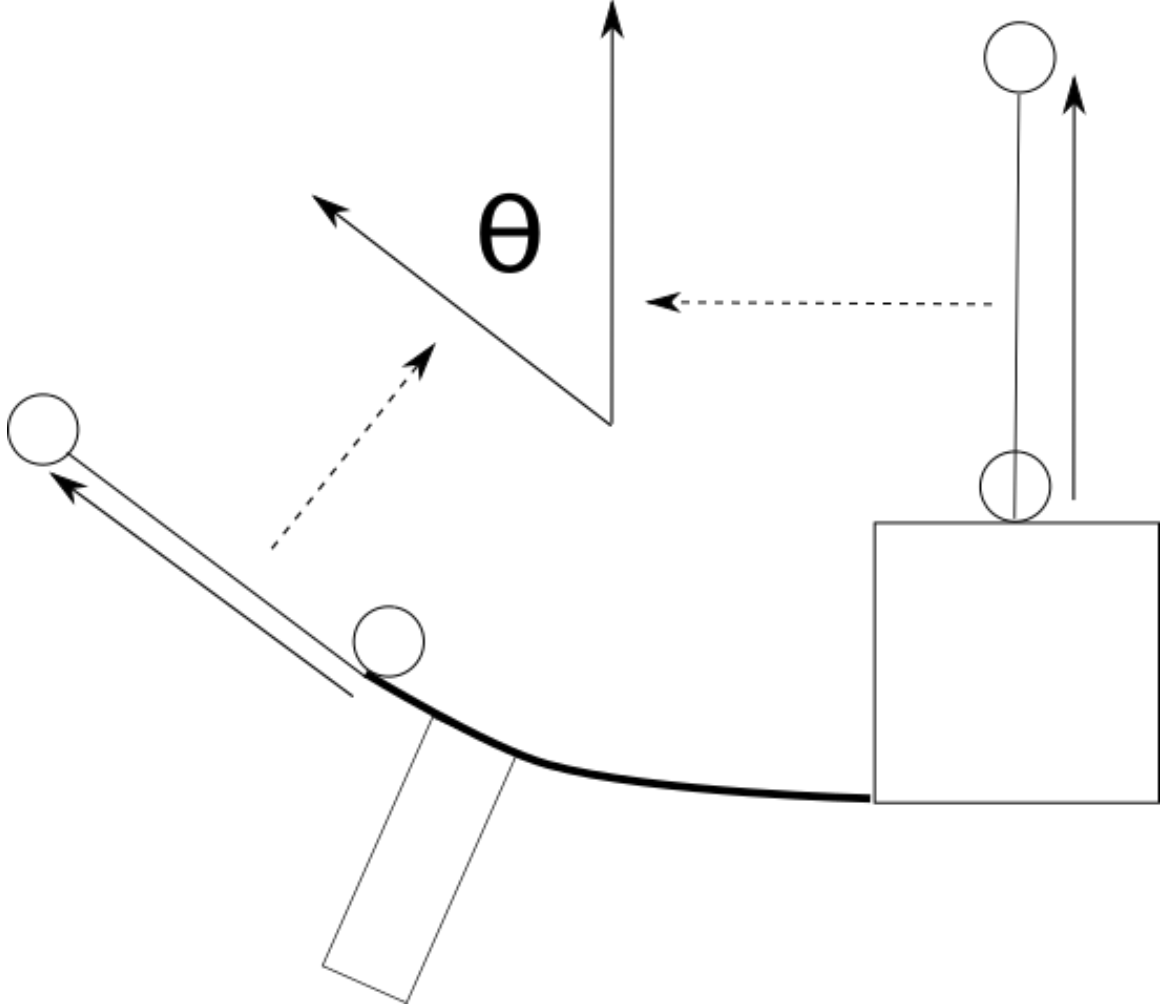


Figure 2.14: A schematic of how the vertical deflection angle is determined from marker data. The distal markers are used as an angular gain to compensate for the resolution of the motion capture system.

$x$  and  $y$  axes correspond to those coordinates. Cooler colors are earlier phase values (closer to 0), while warmer colors are later phase values (closer to  $2\pi$ ). We do not observe a meaningful correlation between the limbs. As such, we instead aggregate data by projecting onto the first two principal axes. We argue that such a reduction is justified based on the singular value spectrum depicted in Fig. 2.18 – the first two singular values summarize the data substantially. We plot the first two principal components of the nominal gait in Fig. 2.19.

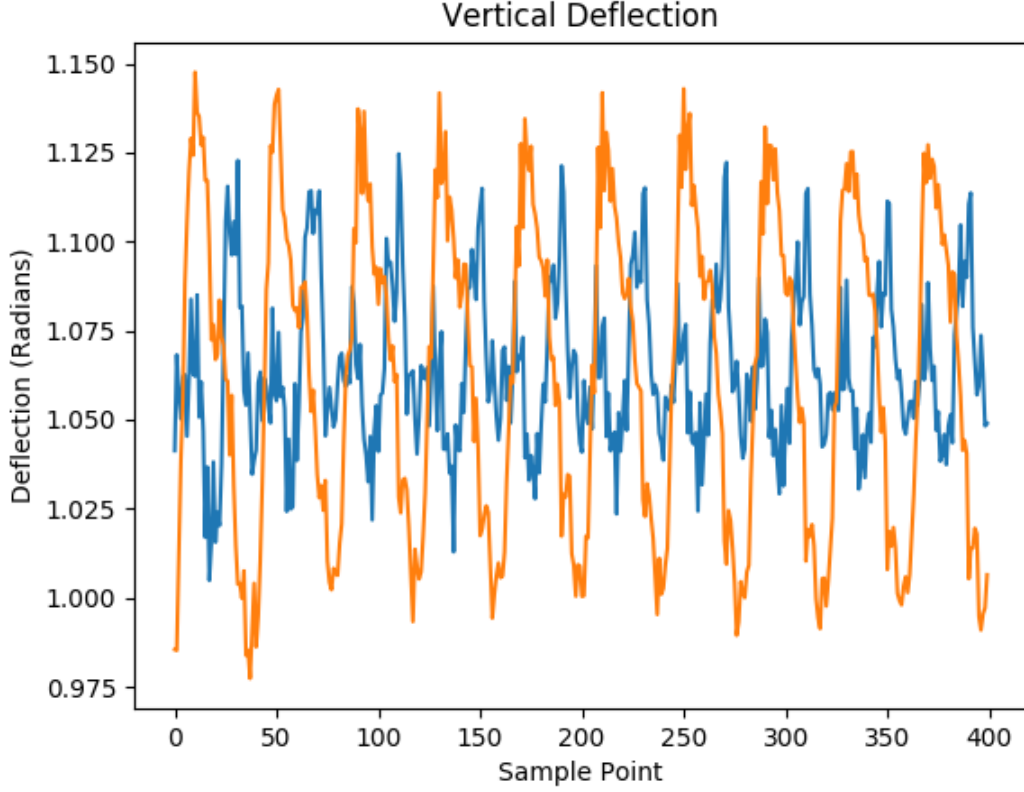


Figure 2.15: Time series data - the (BR,FR,ML) tripod signal is colored orange, while the (BL,MR,FL) tripod signal is shown in blue. The jump near data point 600 is where the robot changed directions.

### 2.6.1.7 Nominal Goal Function

Our goal is to fit the constraints, as mentioned.

$$\tilde{J}[\mu] := \lambda \int_0^T \|\Omega(x) \cdot \dot{x} - \gamma(t, x)\|^2 dt \quad (2.70)$$

Our constraints  $\Omega$  are the derivatives of the vertical and horizontal spring deflection signals, so the cost function takes the explicit form:

$$J(\mu) := \left( \|\dot{V}_i - \dot{V}_{ref}\|^2 + \|\dot{H}_i - \dot{H}_{ref}\|^2 \right) \quad (2.71)$$

We are evaluating this cost function on the noisy data arising from the physical

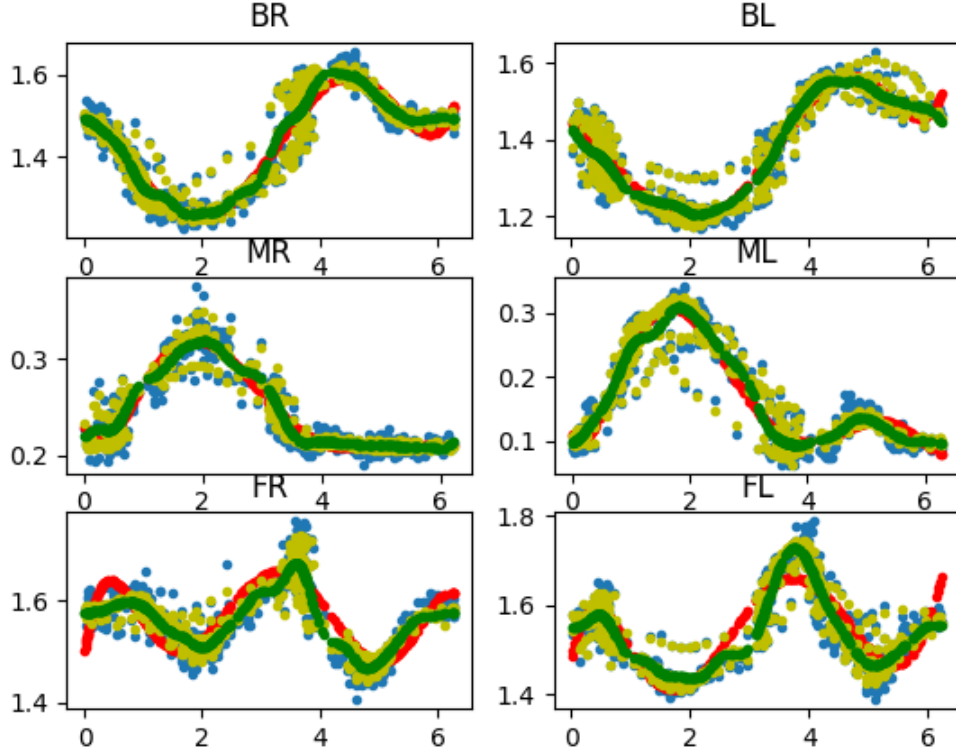


Figure 2.16: Phase plots of the vertical spring deflection. The blue is the original data plotted as a function of phase. Yellow is after running a 2-order 2-sample discrete low pass filter twice, yellow is a 2nd order polynomial fit, while green is a 10th order Fourier series. We see considerable symmetry between tripods. Note that the (BL, MR, FL) tripod has been phase-shifted by  $\pi$  for clarity.

motion of the robot. We mitigate this by collecting enough data so that the distance between means of two parameter sets is greater than the variance of either singular parameter set. Ergo, we always take a window of  $n = 35$  strides (a stride is a single gait period, where phase elapses from 0 to  $2\pi$ ), and average over them to get a mean value-per-stride (where “value” is the value of a function given via context) As such,  $t_0$  and  $t_f$  are fixed at  $t_0 = 0$ ,  $t_f = 35 \times \frac{1}{2.5} = 14$  seconds. Thus, we have operationally evaluate, for  $x_i, y_i, \theta_i$  the relevant values on the  $i$ -th stride,

$$J_0(\mu) := \frac{1}{35} \sum_{x=1}^{35} J(\mu, x_i, y_i, \theta_i) \quad (2.72)$$

### 2.6.2 Results

The optimization was conducted using Nelder-Mead on the four-parameter space  $\mu$  that defines the signal driving the central module. The evolution of the cost is displayed in Fig. 2.22. We terminated the optimization after we had approximately reduced the cost by 40%. Our choice for this was empirically motivated, as we had achieved acceptable performance, as depicted in Fig. 2.21. We admit this termination condition is fairly arbitrary, and future work will include a more principled approach to classifying a “success”.

Modulo the above conditions, we see that even after  $N = 36$  iterations (where a single iteration corresponds to at least 35 strides executed at a set of hyper-parameters  $\mu$ ), the mean displacement per stride is significantly enhanced.

The distribution of points is wider, but we do not necessarily expect that the variance be small. Inspecting the evolution of the cost function, iteration 26 and 33 look like “bad” parameter sets that introduced significant growth in the cost function. Presumably, if this optimization continued until the variability in cost was substantially reduced, the distribution of effective distance per stride would be more narrow.

The point we wish to communicate with our results is that a relatively small number of iterations,  $N = 36$ , along with zero predictive model information, was able to generate a useful forward motion relatively quickly by enforcing empirically determined constraints exclusively with experimental data.

## 2.7 Relationship with Other Learning Perspectives

### 2.7.1 Kinematic Synergies

The method of kinematic synergies is a dimension-reduction approach prominently seen in analysis of human manipulation tasks as a means of reducing the dimension-

ality of DoFs of the hand (*Jarque-Bou et al., 2019, 2016*). A behavioral specification  $(\phi, \Omega, \gamma)$  yields an analogous mechanism to provide a convenient reduction of the number of independent coordinates in  $Q_R$ .

Each constraint  $\Omega_i$  is a row of  $\Omega$ ; by the fundamental theorem of linear algebra, the span of rows of  $\Omega$ , equivalently  $\text{im}(\Omega^T)$ , span the orthogonal complement to the nullspace of  $\Omega$ . Assuming we have a metric, we can consider the covector  $\Omega_i$  to be a tangent vector of  $Q_R$  via the musical isomorphism (*Lee, 2013*), e.g.,  $\Omega_i^T$  is a tangent vector in the standard metric. Assuming the rank of  $\Omega$  is  $n$ , we obtain  $n$  linearly independent vectors that indicate how to “slave” the components of  $\dot{x}$  to efficiently generate trajectories in  $Q_R$  that obey  $\Omega \cdot \dot{x} = \gamma$ ; two immediate consequences follow. The first is that for tangent vectors  $v, w$ , if  $\Omega_i(x) \cdot v = \Omega_i(x) \cdot w$ , then  $\gamma_i$  is the same by definition. Thus, if damage occurs such that  $v$  is no longer possible, the vector  $\Omega_i$  provides explicit knowledge of how to covary the velocity components to produce a  $w$  that preserves  $\gamma_i$ .

The second is to consider a family of behavior specifications that are a result of the same constraint forms, i.e.,  $(\phi, \Omega, \gamma^I)$ , where  $\gamma^I$  is a indexed collection of constraint values  $\gamma$ . Then, we see that the  $\Omega_i^T$ , interpreted as tangent vectors, provide a suitable basis for  $TQ_R$  to naturally implement various choice of  $\gamma^I$ . Thus, the system, for control inputs  $u_i$

$$\dot{x} = \sum_i^n \Omega_i^T(x) u_i \quad (2.73)$$

is able to implement the family  $\gamma^I$  with  $n \leq \dim Q_R$  inputs.

### 2.7.2 Scaffold

Humans exhibit a series of development milestones while learning to walk (*Adolph and Robinson, 2013*). It has been found that incorporating such milestones into a scaffold of nested behaviors can dramatically improve the rate at which robots can iteratively learn complex behaviors (see *Bongard (2011)*, and the references therein).

Since the pullback of a differential form is defined for any full-rank map between manifolds, our approach suggests a natural extension to a scaffold, i.e., we can just as easily have a sequence of encoding templates  $Q_{E_1}, \dots, Q_{E_n}$  and corresponding behavioral specifications  $(\phi_{n-1:n}, \Omega_{E_n}, \gamma_{E_n})$ , where  $\phi_{n-1:n}$  maps template  $E_{n-1}$  into  $E_n$ , with the convention  $E_0 := Q_R$ .

A scaffold like this, where constraints are iteratively pulled back, allows a learning strategy to construct complex behaviors for highly actuated robots out of lower-DoF “sub-behaviors”. Designers, be they engineers or autonomous optimization tools, could initially design a curve  $x(t) \subset E_{n-1}$  that obeys a behavior specification for  $E_n$ . Then, they can use  $x(t)$  to augment an existing behavior specification on  $E_{n-2}$ . In designing a new curve in  $E_{n-2}$ , we gain the ability to both *preserve* the first behavior, while enforcing a new one. We could imagine that the dimension of each template continues to grow through each pullback, allowing us to constructively lift component behaviors into increasingly complex anchors.

### 2.7.3 Planning

A key feature of our encoding templates that removes many difficulties associated with the conventional hypothesis of templates and anchors is that we do not suppose that all curves in the encoding template can be realized as trajectories of the anchor, but only the distinguished  $y(t)$ . Equivalently, we only insist that the constraints are satisfied along the specified  $y(t)$ , rather than everywhere on the encoding template.

A corollary is that the constraints  $\omega_i$  are not suitable for *planning*. If the template constraints corresponded to a kinematic car, the trajectories of the car other than the one output we chose are not required to be achievable by the anchor system. If the  $\omega_i$  are defined over the entirety of  $Q_E$ , they could be used to classify multiple output curves, but we emphasize that this is not the same as requiring *every* output curve to be achievable. Additionally, our constraints  $\omega_i$ , even if derived from a known



model (e.g., a kinematic car) need not agree with the constraint *values* that produce the dynamics that motivated their choice. E.g., by our definition of a behavior specification, we are free to have  $\omega(y) \cdot \dot{y} = \gamma$  for  $\gamma \neq 0$ , even if as designers we know that  $\gamma = 0$  agrees with a known system with Pfaffian constraints. The point being that our construction requires only a specific trajectory of the template system to be followed. If one attempted to generate additional trajectories of the template system to lift, additional care would need to be taken to ensure it is possible to realize them.

#### 2.7.4 Relationship with Output Tracking

A point we wish to emphasize is we are not only choosing an arbitrary output  $y(t)$ , and suggesting that it be stabilized. Doing such a thing is well within the wide literature that exists for output tracking. While our constraint-based optimization function offers an empirically rapid solution technique for such a problem, our contribution is that the constraints  $\omega_i$  *define* the output  $y(t)$ .

We *elected* constraints first, and evaluated them as output functions along a known behavior  $x_0(t)$ , which is how we originally obtained the definition of  $y(t)$ . We then argued re-implementation was the correct strategy. In this, we are arguing that a desired behavior specification can be obtained from a known  $x_0(t)$  in a very cavalier way. Any set of  $\omega$ , and  $\phi$ , that meet our requirements are in principle equally good for defining  $y(t)$ , which is a component of our motivation for omitting  $y$  from the notation  $(\phi, \Omega, \gamma)$ . It also motivates the adjective “encoding” – we have found *a description* of a desired behavior, but there is not a unique encoding.

## 2.8 Summary

We have developed a recovery strategy that represents a desired behavior as a unique solution to a collection of constraint equations on output variables. By doing so, we hope to gain several benefits. The first of which is dimension reduction. The

desired behavior has a *non-trivial* kernel as a result of projecting using the phase-map  $\varphi$ . An entire continuum of solutions for the robot exist in this space, allowing any one of these choices to suffice. Secondly, we demonstrated that while model-based constraints are *sufficient* to characterize desired template behavior, there exists an open and dense subset of observation variables that allow the desired behavior to be completely expressed using *measurements*. We note that we are not necessarily defining first integrals, as the constraints are not required to be invariant.

We also show that while the above works in simulation, it also seems to produce an effective recovery strategy on an experimental system using optimization. Our cost function defined by the constraints discovers effective motion in only 36 iterations – comparatively few. A *rapid* convergence rate of a recovery strategy is an obvious benefit versus one slower, but we would further argue that other attempts to use model-free reinforcement learning strategies on robots have not been as fruitful as one would hope. Our technique eliminates the need to explore a parameter space blindly – directions that are “good” or “bad” are entirely encompassed by the necessary conditions resulting from the constraints. By projecting onto a sub-bundle of the tangent bundle is a sufficient condition for planning compensating inputs, we argue that contrary to “black-box” methods that are obstructed by high-dimension, we can *exploit* high-dimension redundancy.

## 2.9 Future Work

The behavioral specification strategy we have outlined presents several immediate directions for future work. The most proximal would be a comparative analysis to more conventional approaches for hardware-in-the-loop optimization, e.g., terminal cost functions. We would expect our approach to out-perform other model-free strategies as our constraints allow explicit classification of “good” directions and “bad” directions, improving the conditioning of a cost function gradient. In partic-

ular, for very high DoF systems, we would expect significant savings. E.g., while an 18 DoF robot would generally be considered problematically high-dimensional for model-free optimization, we would expect our technique to converge quickly, as with 18 DoFs over six limbs connected to a single torso, the system is highly redundant. Additionally, while model-free optimization immediately suggests shooting methods, since the constraints are *locally* (infinitesimally) defined, direct methods (e.g., collocation) that “stitch” segments of curve together may be particularly appropriate for our strategy. Better understanding algorithmic approaches to solving our proposed constraint problem would be fruitful.

Another possible avenue to pursue is how to account for multiple, competing, behavioral specifications. If the lifted constraints have orthogonal spans, then it is immediate that both could be implemented by the same device. However, a more interesting case is when behavior specifications compete – it may be fruitful to explore how to perform design trade-offs in a formal sense to “best-fit” competing design objectives.

Finally, while we considered the constraints as fixed functions, and the state as mutable, it may instructive to allow the constraints themselves to be directly controlled. More-so than parametric manipulation of constraint coefficients, though that may be of interest, is allowing different learned constraints to be in effect for different regions of statespace. This would allow “hybrid” constraints that can be activated discretely; such a case is known to have interesting geometric properties (*Kelly and Murray, 1995*).

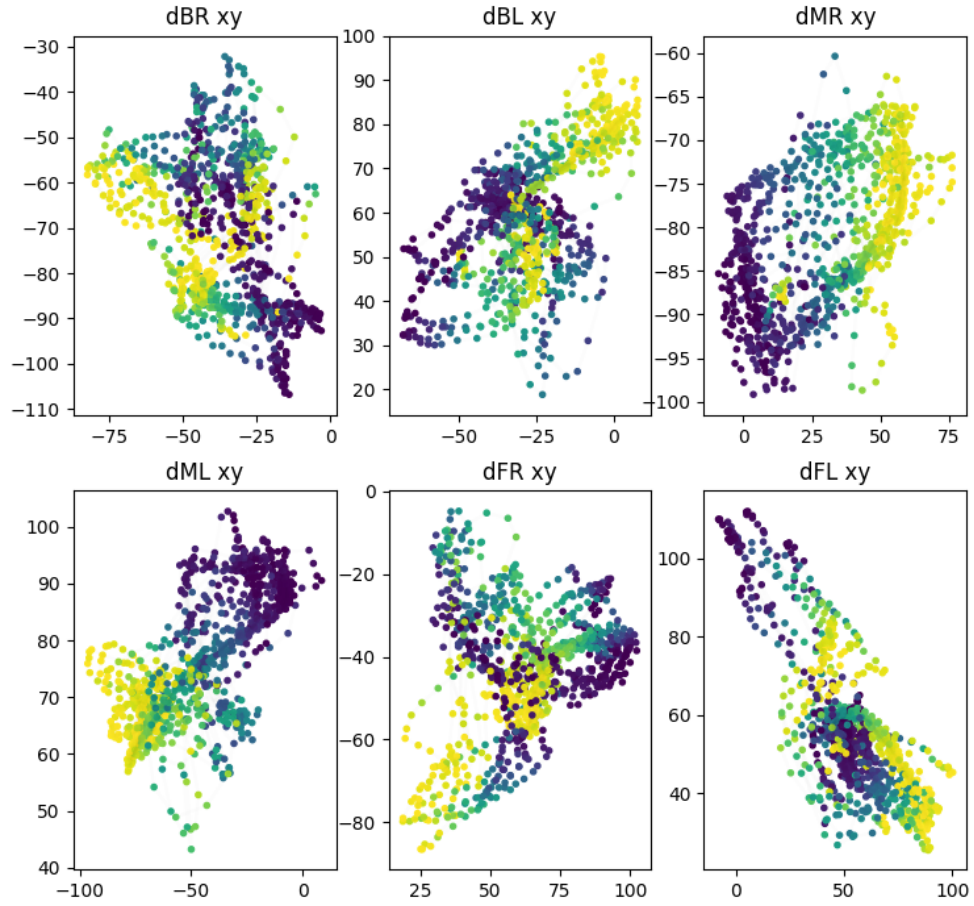


Figure 2.17:  $fa^i(t)$  for each limb, organized by phase. Darker colors are earlier values of phase, while warm colors are later phase values. We observe that there is not an apparent consistency between various limbs.

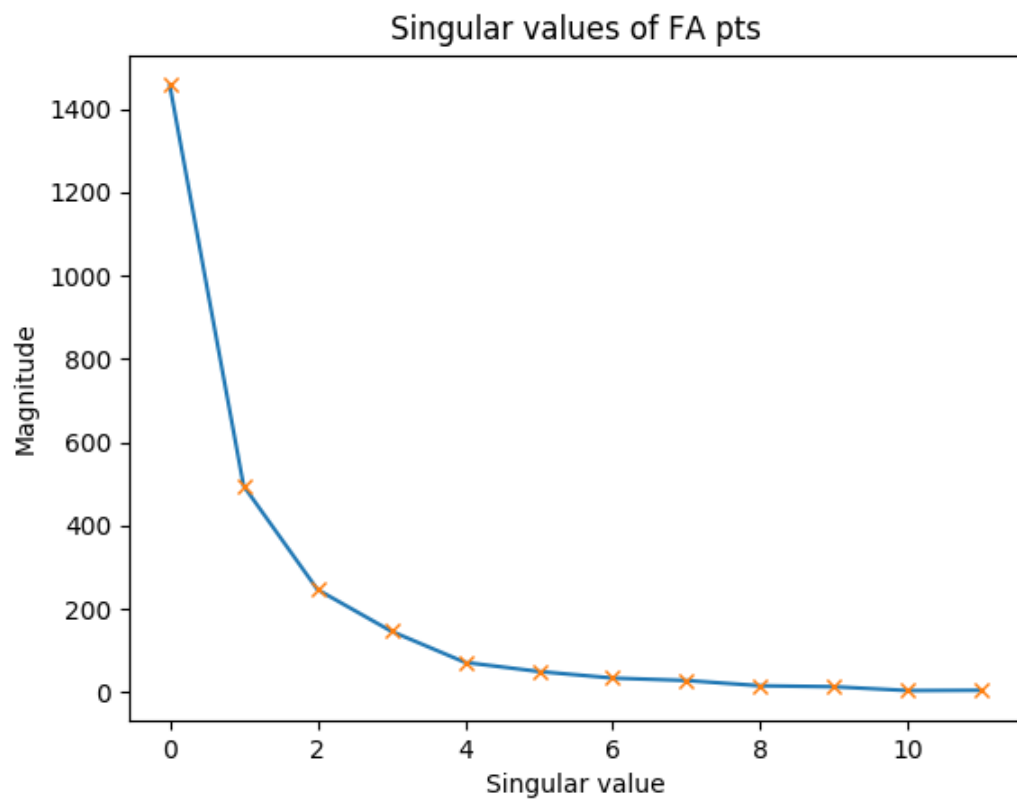


Figure 2.18: The singular values of the  $fa^i$  data. It appears the data is predominately two-dimensional, but in such a way that does not correspond direct deflection of each tripod.

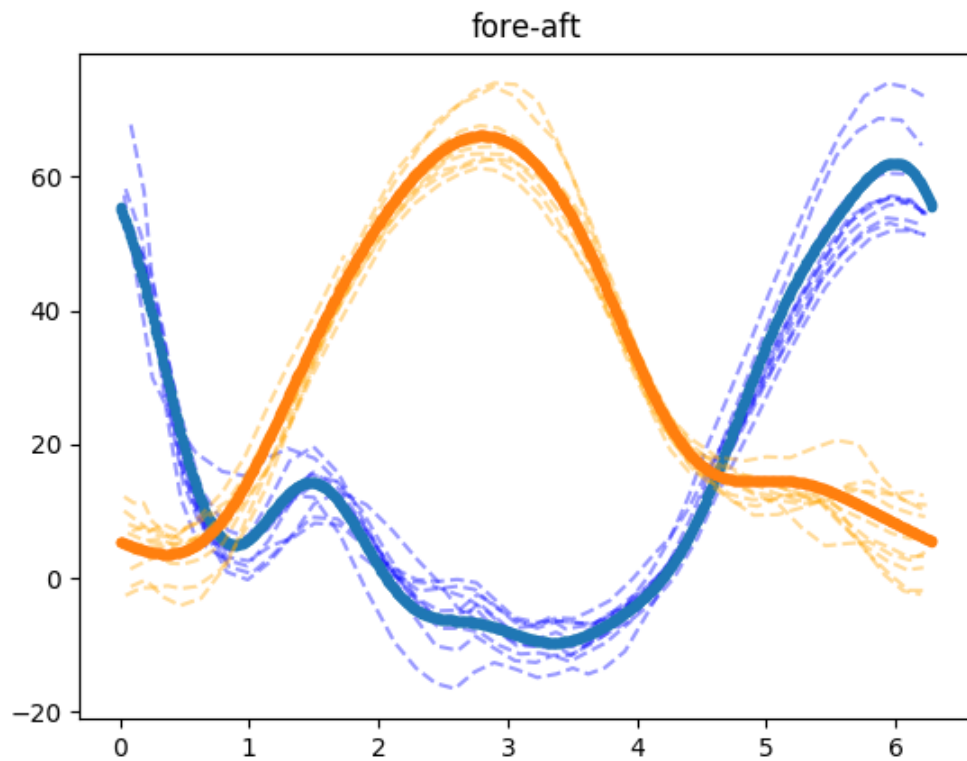


Figure 2.19: The first two principal components of the fore-aft deflection data. The dashed lines indicate measurement data, while the solid line is the Fourier-series approximation of the data.

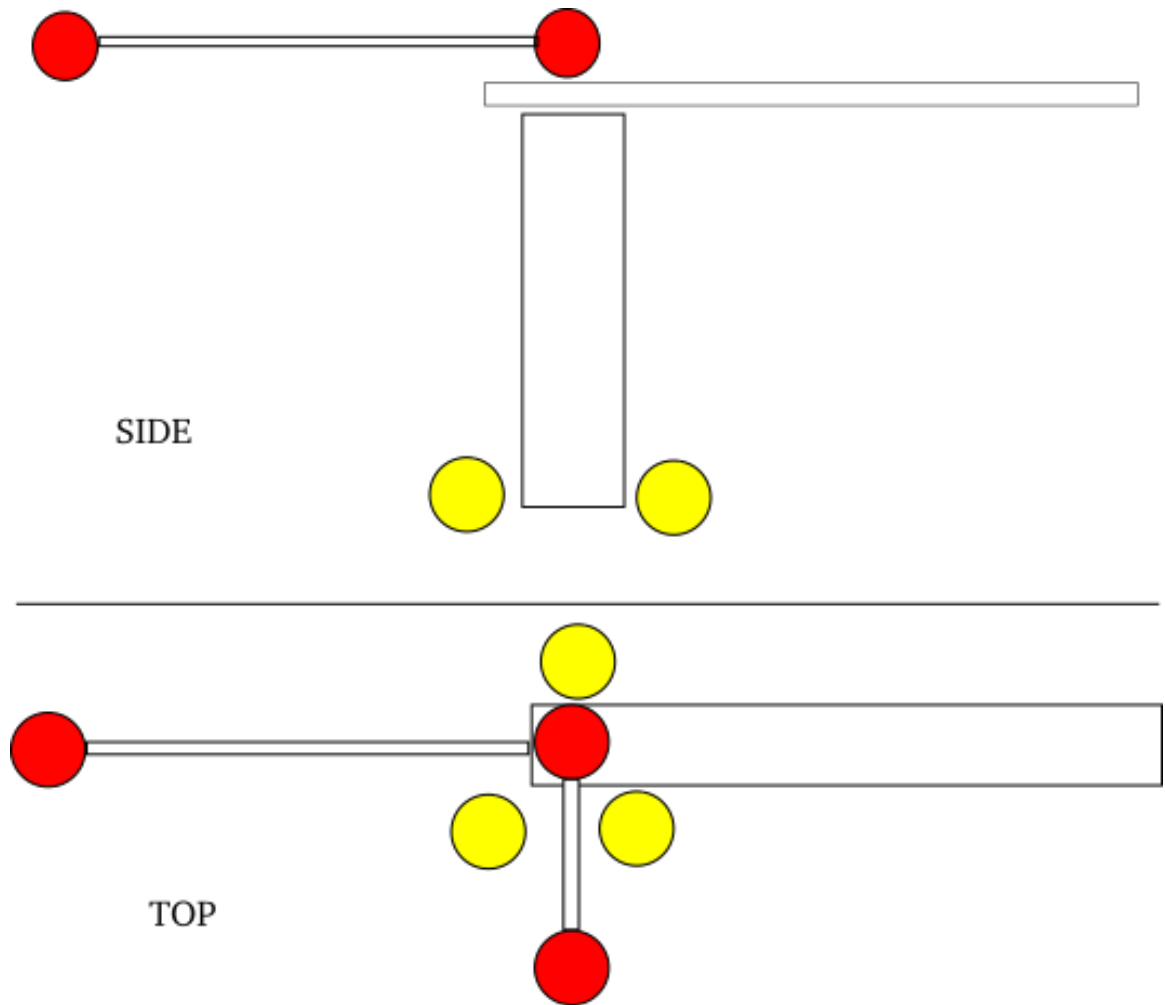


Figure 2.20: Sagittal and aerial view of one leg. The red and yellow groupings of markers are used to determine a relative deflection

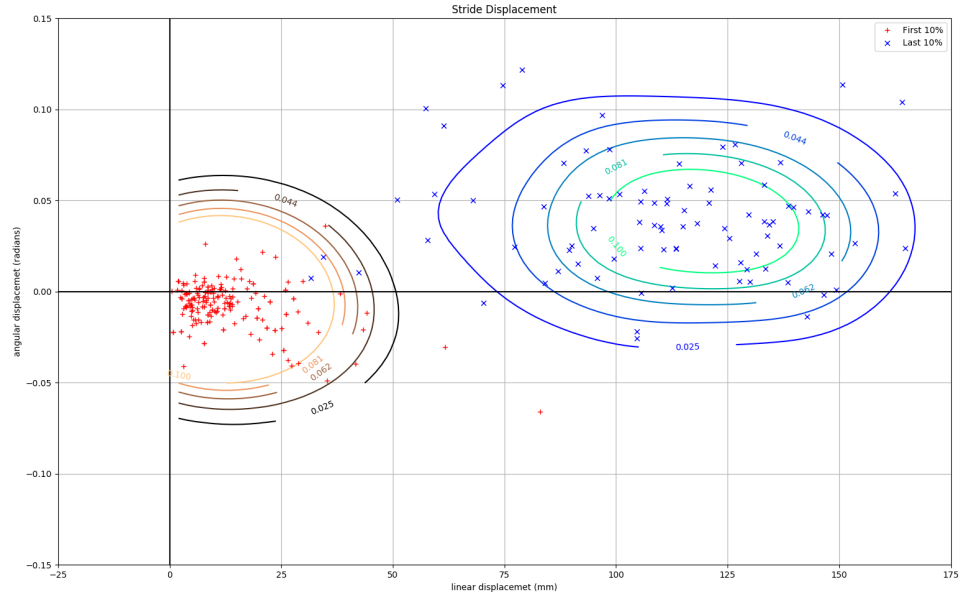


Figure 2.21: A comparison of pre and post optimization. The We the represented  $\xi = (\xi_x, \xi_y, \xi_\theta)$  by the 2D position with  $x = (\xi_x^2 + \xi_y^2)^{1/2}$  and  $y = \xi_\theta \in [0, 2\pi]$ . We plotted the results of individual strides (the red pluses are the first ten percent of strides; the blue “x”’s are the last ten percent of strides; as well as the contours of a kernel smoothed density produced from those data (blue to cyan last points, black to orange the first ten percent of points



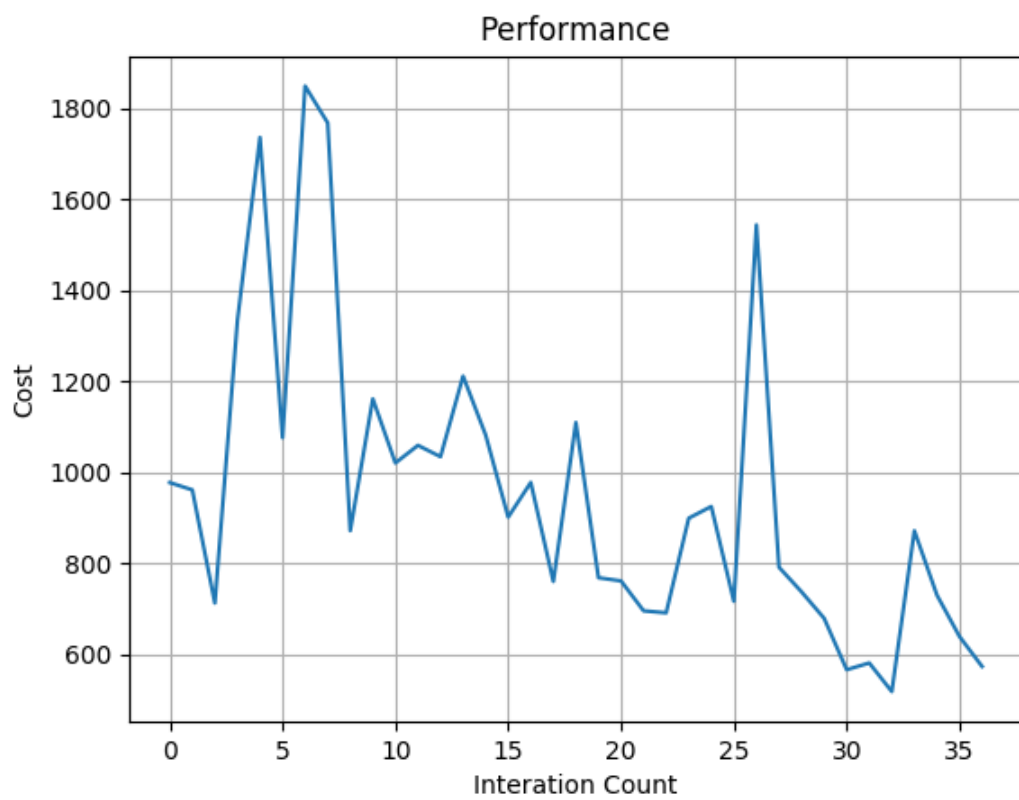


Figure 2.22: The cost function along for the enepod after  $N = 36$  iterations.

## BIBLIOGRAPHY

### Bibliography

(2019), *ATOF Blade Fuses Rated 32*, LittleFuse.

Adolph, K. E., and S. R. Robinson (2013), The road to walking: What learning to walk tells us about development, *Oxford handbook of developmental psychology*, 1, 403–443.

Aizerman, M., and F. Gantmacher (1958), Determination of stability by linear approximation of a periodic solution of a system of differential equations with discontinuous right-hand sides, *The Quarterly Journal of Mechanics and Applied Mathematics*, 11(4), 385–398.

Alexander, K., et al. (1994), *Smooth invariant manifolds and normal forms*, vol. 7, World Scientific.

Alexander, R. M. (1984), The gaits of bipedal and quadrupedal animals, *The International Journal of Robotics Research*, 3(2), 49–59.

Ankarali, M., and U. Saranli (2010), Stride-tostride energy regulation for robust self-stability of torque-actuated dissipative spring-mass hopper, *Chaos*, 20(033121), doi:10.1063/1.3486803.

Arnold, V. (1973), *Ordinary differential equations*, The MIT Press, Cambridge, Massachusetts.

Arslan, O., and U. Saranli (2012), Reactive planning and control of planar spring-mass running on rough terrain, *IEEE Trans. Robotics*, 28, 567–579, doi:10.1109/TRO.2011.2178134.

Bellman, R. E. (1961), Dynamic programming treatment of the traveling salesman problem.

- Bernardo, M., C. Budd, A. R. Champneys, and P. Kowalczyk (2008), *Piecewise-smooth dynamical systems: theory and applications*, vol. 163, Springer Science & Business Media.
- Bizzarri, F., A. Brambilla, and G. S. Gajani (2013), Lyapunov exponents computation for hybrid neurons, *Journal of computational neuroscience*, *35*(2), 201–212.
- Blickhan, R. (1989a), The spring-mass model for running and hopping, *J. Biomechanics*, *22*(11-12), 1217–1227, doi:10.1016/0021-9290(89)90224-8.
- Blickhan, R. (1989b), The spring-mass model for running and hopping, *J. Biomechanics*, *22*(11-12), 1217–1227, doi:10.1016/0021-9290(89)90224-8.
- Bloch, A., J. Baillieul, P. Crouch, J. E. Marsden, D. Zenkov, P. S. Krishnaprasad, and R. M. Murray (2003), *Nonholonomic mechanics and control*, vol. 24, Springer.
- Bloch, A. M., P. Krishnaprasad, J. E. Marsden, and R. M. Murray (1996), Nonholonomic mechanical systems with symmetry, *Archive for Rational Mechanics and Analysis*, *136*(1), 21–99.
- Bogert, A., K. Gerritsen, and G. Cole (1998), Human musclemodelling from a user’s perspective, *J. Elecromyogr. Kines.*, *8*(2), 119–124, doi:10.1016/S1050-6411(97)00028-X.
- Bongard, J., V. Zykov, and H. Lipson (2006), Resilient machines through continuous self-modeling, *Science*, *314*(5802), doi:10.1126/science.1133687.
- Bongard, J. C. (2011), Morphological and environmental scaffolding synergize when evolving robot controllers: artificial life/robotics/evolvable hardware, in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pp. 179–186, ACM.
- Browder, F. E. (1954), Covering spaces, fibre spaces, and local homeomorphisms, *Duke Math. J.*, *21*(2), 329–336, doi:10.1215/S0012-7094-54-02132-8.
- Bullo, F., A. D. Lewis, and K. M. Lynch (2002), Controllable kinematic reductions for mechanical systems: concepts, computational tools, and examples, in *Mathematical Theory of Networks and Systems*, vol. 124.
- Burden, S. A., S. Revzen, and S. S. Sastry (2015), Model reduction near periodic orbits of hybrid dynamical systems, *IEEE Transactions on Automatic Control*, *60*(10), 2626–2639.
- Burden, S. A., S. S. Sastry, D. E. Koditschek, and S. Revzen (2016), Event-selected vector field discontinuities yield piecewise-differentiable flows, *SIAM Journal on Applied Dynamical Systems*, *15*(2), 1227–1267.
- Carver, S., N. Cowan, and J. Guckenheimer (2009), Lateral stability of the spring-mass model suggests a two-step control strategy for running, *CHAOS*, *19*, doi:10.1063/1.3127577.

- Council, G., S. Yang, and S. Revzen (2014), Deadbeat control with (almost) no sensing in a hybrid model of legged locomotion, in *Advanced Mechatronic Systems (ICAMEchS), 2014 International Conference on*, pp. 475–480, IEEE.
- Cully, A., J. Clune, D. Tarapore, and J. Mouret (2014), Robots that can adapt like animals, *Nature*, *521*(7553), doi:10.1038/nature14422.
- Da, X., O. Harib, R. Hartley, B. Griffin, and J. W. Grizzle (2016), From 2d design of underactuated bipedal gaits to 3d implementation: Walking with speed tracking, *IEEE Access*, *4*, 3469–3478.
- Deisenroth, M. P., G. Neumann, J. Peters, et al. (2013), A survey on policy search for robotics, *Foundations and Trends® in Robotics*, *2*(1–2), 1–142.
- Dieci, L., and L. Lopez (2011), Fundamental matrix solutions of piecewise smooth differential systems, *Mathematics and Computers in Simulation*, *81*(5), 932–953.
- Ernst, M., H. Geyer, and R. Blickhan (2011), Spring-legged locomotion on uneven ground: A control approach to keep the running speed constant, in *Int. Conf. on Climbing and Walking Robots*, pp. 639–644.
- Farley, C., R. Blickhan, and C. Taylor (1985), Mechanics of human hopping : model and experiments, *Am. Zool.*, *25*, 54A.
- Fenichel, N. (1974), Asymptotic stability with rate conditions for dynamical systems, *Bulletin of the American Mathematical Society*, *80*(2), 346–349.
- Fenichel, N. (1977), Asymptotic stability with rate conditions, ii, *Indiana University Mathematics Journal*, *26*(1), 81–93.
- Filippov, A. F. (2013), *Differential equations with discontinuous righthand sides: control systems*, vol. 18, Springer Science & Business Media.
- Full, R. J., and D. E. Koditschek (1999), Templates and anchors: neuromechanical hypotheses of legged locomotion on land, *Journal of experimental biology*, *202*(23), 3325–3332.
- Galewski, M., and M. Koniorczyk (2016), *Global invertibility theorems and their applications-a variational approach*, Wydawnictwo Politechnika Łódzka.
- Galloway, K. C., G. C. Haynes, B. D. Ilhan, A. M. Johnson, R. Knopf, G. Lynch, B. Plotnick, M. White, and D. E. Koditschek (2010), X-rhex: A highly mobile hexapedal robot for sensorimotor tasks, *Tech. rep.*, University of Pennsylvania.
- Ghigliazza, R., R. Altendorfer, P. Holmes, and D. Koditschek (2004), A simply stabilized running model, *SIAM J. App. Dyn. Systems*, *2*(2), 187–218, doi:10.1137/S1111111102408311.

- Ghigliazza, R. M., and P. Holmes (2005), Towards a neuromechanical model for insect locomotion: Hybrid dynamical systems, *Regular & Chaotic Dynamics*, 10(2), 193–225, doi:10.1070/RD2005v010n02ABEH000311.
- Goebel, R., R. Sanfelice, and A. Teel (2009), Hybrid dynamical systems, *Control Systems, IEEE*, 29(2), 28–93, doi:10.1109/MCS.2008.931718.
- Golubitsky, M., and V. Guillemin (2012), *Stable mappings and their singularities*, vol. 14, Springer Science & Business Media.
- Groff, R. E., P. P. Khargonekar, and D. E. Koditschek (2003), A local convergence proof for the minvar algorithm for computing continuous piecewise linear approximations, *SIAM journal on numerical analysis*, 41(3), 983–1007.
- Guckenheimer, J. (), Isochrons and phaseless sets, *Journal of Mathematical Biology*, 1(3), 259–273, doi:10.1007/BF01273747.
- Guckenheimer, J., and S. Johnson (1995), *Hybrid systems II*, 202–225 pp., Springer-Verlag, London, UK.
- Gupta, K., and A. P. Pobil (1998), *Practical motion planning in robotics: Current approaches and future directions*, John Wiley & Sons, Inc.
- Hairer, E., G. Wanner, and P. Norsett (1993), Solving ordinary differential equations i – nonstiff problems, Springer Series in Computational Mathematics, 2 ed., Springer-Verlag.
- Hatton, R. L., and H. Choset (2011), An introduction to geometric mechanics and differential geometry.
- Hatton, R. L., and H. Choset (2013), Geometric swimming at low and high reynolds numbers, *IEEE Transactions on Robotics*, 29(3), 615–624.
- Hereid, A., E. A. Cousineau, C. M. Hubicki, and A. D. Ames (2016), 3d dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics, in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pp. 1447–1454, IEEE.
- Hill, A. (1938), The heat of shortening and the dynamic constants of muscle, *Proc. R. Soc. Lond. B*, 126(843), 136–195, doi:10.1098/rspb.1938.0050.
- Hirsch, M. W., R. L. Devaney, and S. Smale (1974), *Differential equations, dynamical systems, and linear algebra*, vol. 60, Academic press.
- Hirsch, M. W., C. C. Pugh, and M. Shub (2006), *Invariant manifolds*, vol. 583, Springer.
- Hirsch, M. W., S. Smale, and R. L. Devaney (2012), *Differential equations, dynamical systems, and an introduction to chaos*, Academic press.

- Holmes, P., R. Full, D. Koditschek, and J. Gukenheimer (2006a), The dynamics of legged locomotion : models, analyses, and challenges, *SIAM Review*, 48(2), 206–304, doi:10.1137/S0036144504445133.
- Holmes, P., R. Full, D. Koditschek, and J. Gukenheimer (2006b), The dynamics of legged locomotion : models, analyses, and challenges, *SIAM Review*, 48(2), 206–304, doi:10.1137/S0036144504445133.
- Hutter, M., R. C.D. M. Hoepflinger, and R. Seigwart (2010), Full state control of a slip model by touchdown detection, in *13th International Conference on Climbing and Walking Robots*, pp. 533–540.
- Ivanov, A. (1998), The stability of periodic solutions of discontinuous systems that intersect several surfaces of discontinuity, *Journal of Applied Mathematics and Mechanics*, 62(5), 677–685.
- Jarque-Bou, N., V. Gracia-Ibáñez, J.-L. Sancho-Bru, M. Vergara, A. Pérez-González, and F. Andrés (2016), Using kinematic reduction for studying grasping postures. an application to power and precision grasp of cylinders, *Applied ergonomics*, 56, 52–61.
- Jarque-Bou, N. J., A. Scano, M. Atzori, and H. Müller (2019), Kinematic synergies of hand grasps: a comprehensive study on a large publicly available dataset, *Journal of neuroengineering and rehabilitation*, 16(1), 63.
- Kelly, S. D., and R. M. Murray (1995), Geometric phases and robotic locomotion, *Journal of Robotic Systems*, 12(6), 417–431.
- Kenneally, G., A. De, and D. E. Koditschek (2016), Design principles for a family of direct-drive legged robots, *IEEE Robotics and Automation Letters*, 1(2), 900–907.
- Khalil, H. (2002a), *Non-Linear System: Third Edition*, Prentice Hall, Upper Saddle River, New Jersey.
- Khalil, H. (2002b), *Nonlinear systems*, 3 ed., Prentice Hall, Upper Saddle River, New Jersey.
- Kobayashi, S., and K. Nomizu (1963), *Foundations of differential geometry*, vol. 1, New York.
- Kober, J., J. A. Bagnell, and J. Peters (2013), Reinforcement learning in robotics: A survey, *The International Journal of Robotics Research*, 32(11), 1238–1274.
- Koditschek, D. E., and E. Rimon (1990), Robot navigation functions on manifolds with boundary, *Advances in applied mathematics*, 11(4), 412–442.
- Koon, W. S., and J. E. Marsden (1997), The geometric structure of nonholonomic mechanics, in *Proceedings of the 36th IEEE Conference on Decision and Control*, vol. 5, pp. 4856–4861, IEEE.

- Kuhn, H. W. (1960), Some combinatorial lemmas in topology, *IBM Journal of research and development*, 4(5), 518–524.
- Lee, J. M. (2013), Smooth manifolds, in *Introduction to Smooth Manifolds*, pp. 1–31, Springer.
- Levine, S., P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen (2018), Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection, *The International Journal of Robotics Research*, 37(4-5), 421–436.
- Lygeros, J., K. H. Johansson, S. N. Simic, J. Zhang, and S. S. Sastry (2003), Dynamical properties of hybrid automata, *IEEE Transactions on automatic control*, 48(1), 2–17.
- Marsden, J. E., R. Montgomery, and T. S. Ratiu (1990), *Reduction, symmetry, and phases in mechanics*, vol. 436, American Mathematical Soc.
- Meigniez, G. (2002), Submersions, fibrations and bundles, *Transactions of the American Mathematical Society*, 354(9), 3771–3787.
- Mordatch, I., Z. Popović, and E. Todorov (2012a), Contact-invariant optimization for hand manipulation, in *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*, pp. 137–144, Eurographics Association.
- Mordatch, I., E. Todorov, and Z. Popović (2012b), Discovery of complex behaviors through contact-invariant optimization, *ACM Transactions on Graphics (TOG)*, 31(4), 43.
- Müller, P. C. (1995), Calculation of lyapunov exponents for dynamic systems with discontinuities, *Chaos, Solitons & Fractals*, 5(9), 1671–1681.
- Murray, R., S. Sastry, and Z. Li (1994), *A mathematical introduction to robotic manipulation*, 1 ed., CRC Press.
- Murray, R. M. (2017), *A mathematical introduction to robotic manipulation*, CRC press.
- OLECH, C. (1998), On the wazewski equation, *ZESZYTY NAUKOWE-UNIWERSYTETU JAGIELLONSKIEGO-ALL SERIES-*, 1223, 55–64.
- Orhon, H. E. (2018), Model-based identification and control of a one-legged hopping robot, *arXiv preprint arXiv:1802.09634*.
- Ostrowski, J., and J. Burdick (1998), The geometric mechanics of undulatory robotic locomotion, *The international journal of robotics research*, 17(7), 683–701.
- Ostrowski, J. P. (1996), The mechanics and control of undulatory robotic locomotion, Ph.D. thesis, California Institute of Technology.

- Palmer III, L. R., and C. Eaton (2014), Periodic spring-mass running over uneven terrain using feedforward control of landing conditions, *Bioinspiration and Biomimetics*, 9(3), doi:10.1088/1748-3182/9/036018.
- Pepy, R., A. Lambert, and H. Mounier (2006), Path planning using a dynamic vehicle model, in *2006 2nd International Conference on Information & Communication Technologies*, vol. 1, pp. 781–786, IEEE.
- Pepyne, D. L., and C. G. Cassandras (2000), Optimal control of hybrid systems in manufacturing, *Proceedings of the IEEE*, 88(7), 1108–1123.
- Pratihari, D. K., K. Deb, and A. Ghosh (2002), Optimal path and gait generations simultaneously of a six-legged robot using a ga-fuzzy approach, *Robotics and Autonomous Systems*, 41(1), 1–20.
- Radford, J., and J. Burdick (1998), Local motion planning for nonholonomic control systems evolving on principal bundles, *A, A, 1, 3*.
- Ramezani, A., J. W. Hurst, K. A. Hamed, and J. W. Grizzle (2014), Performance analysis and feedback control of atrias, a three-dimensional bipedal robot, *Journal of Dynamic Systems, Measurement, and Control*, 136(2), 021,012.
- Ratliff, N., M. Zucker, J. A. Bagnell, and S. Srinivasa (2009), Chomp: Gradient optimization techniques for efficient motion planning, in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pp. 489–494, IEEE.
- Reinkensmeyer, D. J., et al. (2014), Tools for understanding and optimizing robotic gait training.
- Reist, P., and R. D’Andrea (2012), Design and analysis of a blind juggling robot, *IEEE Transactions on Robotics*, 28(6), 1228–1243.
- Revzen, S., and J. Guckenheimer (2008), Estimating phase of synchronized oscillators, *Phys. Rev. E*, 78(5), 051,907, doi:10.1103/PhysRevE.78.051907.
- Revzen, S., and M. Kvalheim (2015), Data driven models of legged locomotion, in *SPIE Defense+ Security*, pp. 94,671V–94,671V, International Society for Optics and Photonics.
- Revzen, S., D. E. Koditschek, and R. J. Full (2009), Towards testable neuromechanical control architectures for running, in *Progress in Motor Control*, pp. 25–55, Springer.
- Revzen, S., M. Kvalheim, S. Wilshon, and J. Guckenheimer (2018), Estimating phase from observed trajectories, *In preparation*, pp. 1–20.
- Rockafellar, R. T. (2003), A property of piecewise smooth functions, *Computational Optimization and Applications*, 25(1-3), 247–250.
- Ruff, C. (2002), Variation in human body size and shape, *Annual Review of Anthropology*, 31, 211–232, doi:10.1146/annurev.anthro.31.040402.085407.



- Rumbaugh, J., I. Jacobson, and G. Booch (2004), *Unified modeling language reference manual, the*, Pearson Higher Education.
- Saranli, U., M. Buehler, and D. E. Koditschek (2001), Rhex: A simple and highly mobile hexapod robot, *The International Journal of Robotics Research*, 20(7), 616–631.
- Sastry, S. (1999), *Nonlinear systems : analysis, stability, and control*, Springer.
- Schmitt, J., and P. Holmes (2000), Mechanical models for insect locomotion: dynamics and stability in the horizontal plane i. theory, *Biological cybernetics*, 83(6), 501–515.
- Scholtes, S. (2012), *Introduction to piecewise differentiable equations*, Springer Science & Business Media.
- Seyfarth, A., H. Geyer, M. Gunther, and R. Blickhan (2002), A movement criterion for running, *J. Biomechanics*, 35, 649–655, doi:10.1016/S0021-9290(01)00245-7.
- Seyfarth, A., H. Geyer, and H. Herr (2006), Swing-leg retraction : a simple control model for stable running, *J. of Exp. Bio.*, 206, 2547–2555, doi:10.1242/jeb.00463.
- Sharir, M. (1989), Algorithmic motion planning in robotics, *Computer*, 22(3), 9–19.
- Siepel, J., and P. Holmes (2007), A simple model for clock-actuated legged locomotion, *Regular and Chaotic Dynamics*, 12(5), 502–520, doi:10.1134/S1560354707050048.
- Simić, S. N., K. H. Johansson, S. Sastry, and J. Lygeros (2000), Towards a geometric theory of hybrid systems, in *International Workshop on Hybrid Systems: Computation and Control*, pp. 421–436, Springer.
- Spivak, M. (1965), *Calculus on Manifolds*, Perseus Books Publishing, L. L. C.
- Spröwitz, A., A. Tuleu, M. Vespignani, M. Ajallooeian, E. Badri, and A. J. Ijspeert (2013), Towards dynamic trot gait locomotion: Design, control, and experiments with cheetah-cub, a compliant quadruped robot, *The International Journal of Robotics Research*, 32(8), 932–950.
- Tassa, Y., T. Erez, and E. Todorov (2012), Synthesis and stabilization of complex behaviors through online trajectory optimization, in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4906–4913, IEEE.
- Tedrake, R., T. W. Zhang, and H. S. Seung (2004), Stochastic policy gradient reinforcement learning on a simple 3d biped, in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3, pp. 2849–2854, IEEE.
- Utkin, V. (1977), Variable structure systems with sliding modes, *IEEE Transactions on Automatic control*, 22(2), 212–222.

- Vejdani, H., Y. Blum, M. Daley, and J. Hurst (2013), Bio-inspired swing leg control for spring-mass robots running on ground with unexpected height disturbance, *Bioinspiration and Biomimetics*, 8(4), doi:10.1088/1748-3182/8/4/046006.
- Westervelt, E. R., J. W. Grizzle, and D. E. Koditschek (2003), Hybrid zero dynamics of planar biped walkers, *IEEE transactions on automatic control*, 48(1), 42–56.
- Winters, J. (1990), Hill-based muscle models: a systems engineering perspective, in *Multiple Muscle Systems*, edited by J. Winters and S.-Y. Woo, pp. 69–93, Springer New York, doi:10.1007/978-1-4613-9030-5\_5.
- Yosinski, J., J. Clune, D. Hidalgo, S. Nguyen, J. C. Zagal, and H. Lipson (2011), Evolving robot gaits in hardware: the hyperneat generative encoding vs. parameter optimization., in *ECAL*, pp. 890–897.
- Yu, H., M. Li, W. Guo, and H. Cai (2012), Stance control of the slip hopper with adjustable stiffness of leg spring, in *Mechatronics and Automation (ICMA), 2012 International Conference on*, pp. 2007–2012, IEEE.

## CHAPTER III

# Event Selected Control and Analysis

### 3.1 Introduction

Running or walking legged robots make rhythmic and frequent contact with the ground when executing a periodic gait ([Alexander, 1984](#)), a cyclic collection of time-ordered postures. In order to better understand the dynamic properties of gaits, and to facilitate the design of legged robots, we would like descriptive mathematical models amenable to analysis and computation. For this purpose, we take a gait to be encoded as a stable periodic solution of a dynamical system. The body-limb mechanics that arise from cyclic limb motion are often taken to be piece-wise smooth, where the smoothness of the description is lost due to the representation of contacts ([Holmes et al., 2006b](#)). For each given leg arrangement, where some number of legs are fixed to be in contact, and the others aerial, the governing equations are smooth. As the feet or limbs make or lose contact with the ground, the dynamics induced by contacts discretely change, possibly with simultaneous contacts of limbs with the ground ([Alexander, 1984](#)), causing the governing equations to discontinuously jump from one set to another.

Piecewise-smooth dynamical systems, which switch between smooth vector fields when an execution arrives at discrete submanifolds (so called “guards”), can be considered a type of hybrid system ([Goebel et al., 2009](#)), which feature mixed interactions

between continuous and discrete modes. Dynamical systems with discontinuous vector fields feature considerably increased complexity compared to the classical smooth case, rich with phenomena that do not occur in smooth systems. Features such as sliding modes ([Utkin, 1977](#)), non-determinism ([Lygeros et al., 2003](#)) and Zeno executions ([Simić et al., 2000](#)). Despite this, existence and uniqueness of solutions, continuity with respect to initial conditions, Lyapunov-like arguments, etc. have received much attention from the scientific and engineering community, as they arise in many problems outside of locomotion ([Goebel et al., 2009](#)), including power systems ([Guckenheimer and Johnson, 1995](#)), neurological modeling ([Bizzarri et al., 2013](#)), and manufacturing ([Pepyne and Cassandras, 2000](#)).

For the case of non-intersecting switching surfaces, the existence of a continuous flow ([Filippov, 2013](#); [Bernardo et al., 2008](#)), first-order approximations ([Dieci and Lopez, 2011](#); [Bizzarri et al., 2013](#)), and stability ([Aizerman and Gantmacher, 1958](#); [Lygeros et al., 2003](#)) are well-studied. Given the ubiquity of simultaneous or near-simultaneous touchdown and lift-off events in legged locomotion ([Alexander, 1984](#)), an important case that has received less attention is where the transition surfaces that govern the active smooth vector field intersect. For the special case where there are exactly two transversely intersecting transition surfaces, it has been shown that the flow is continuous, that variational equations producing first-order approximations exist, and that stability criterion exist for periodic solutions. ([Dieci and Lopez, 2011](#); [Ivanov, 1998](#))

In many of the proofs of the cited work, the class of hybrid system is restricted implicitly. In this regard, it can be difficult to evaluate if a given system satisfies the necessary assumptions for a theorem to hold. More recently, a new class of piecewise-smooth differential equations, “event-selected vector fields”, was defined and developed in [Burden et al. \(2016\)](#). Within, a positively-defined class of piecewise-differentiable systems, which includes the general case of an arbitrary number of

switching surfaces, was shown to have a piecewise differentiable flow, admit higher order approximations, and to possess a collection of stability criteria.

In this chapter, we will employ these “event-selected systems” in the context of robotics. A useful feature of said systems is that they possess a first-order approximation akin to a classic derivative. We will introduce a numerical method to compute this approximation. Importantly, compared to a naive approach, we will show our method eliminates the factorial complexity that has traditionally been a computational obstacle associated with multi-contact problems. Our method both reduces the number of integrations needed, and simplifies the complexity of each integration. Even for low-dimensional problems, we see run-times improve by a factor of several hundred.

We will also demonstrate that event selected vector fields predict the existence of a novel type of autonomous control that can stabilize a desired trajectory in a highly distributed manner. We then present an algorithm that allows efficient implementation of this strategy in a distributed way, with robust timing characteristics that do not require the high-bandwidth feedback of conventional closed-loop control.

Since our algorithm is custom and not easily adapted to off-the-shelf hardware, we will conclude with a RHex-type robot ([Saranli et al., 2001](#)) with completely custom hardware that implements our strategy.

## 3.2 Event Selected Systems

Since the results of the sequel depend critically on familiarity with event-selected systems theory, and it is not broadly known, a few key components will be reproduced here. The mathematical constructions of [Burden et al. \(2016\)](#) extensively employ piecewise differentiable calculus, as presented in [Scholtes \(2012\)](#), which generalizes many of the familiar properties of classical calculus.

### 3.2.0.1 Piecewise Differentiable Calculus

The following is a restatement of key definitions and results in [Scholtes \(2012\)](#), which contains comprehensive definitions and proofs for the interested reader. Let  $U$  be an open subset of  $\mathbb{R}^n$  (we will always assume the standard metric topology), and let  $f_i : U_i \rightarrow \mathbb{R}^n \in \mathcal{C}^r(U_i, \mathbb{R}^n)$  be a finite collection of functions. A function  $f : D \rightarrow \mathbb{R}^n$  is called  $\text{PC}^r$  if  $f$  is continuous, and for every  $x_0 \in D$ , there exists an open neighborhood  $U \subset D$ ,  $x_0 \in U$ , such that  $\forall x \in U$ ,  $f(x) \in \{f_1(x), \dots, f_n(x)\}$ . In which case,  $f$  is said to be a *continuous selection* of the *selection functions*  $f_1, \dots, f_k$ . It is assumed that the individual domains of the selection functions have a mutual domain where  $f$  is defined, i.e.  $D \subset \bigcap_i U_i$ . A selection function  $f_i$  is said to be *active* if  $f(x) = f_i(x)$ . In [Rockafellar \(2003\)](#), it is shown that this notion of piecewise smoothness is in fact equivalent to any point  $x \in U$  having a neighborhood covered by a finite number of closed sets  $\{P_j\}_{j \in \mathcal{J}}$ ,  $P_j = \text{Cl}(\text{Int}(P_j))$  such that  $x \in P_j$ ,  $f$  is smooth on  $\text{Int}(P_j)$ , and  $\nabla f$  can be extended continuously to  $\partial P_j$ , which coincides with the intuitive notion of  $f$  consisting of a finite number of smooth pieces.  $\text{PC}^r$  functions are closed under composition, as well as the operations of pointwise max and min; additionally,  $\text{PC}^r$  functions are locally Lipschitz on a compact convex set  $V$  with constant  $L = \max\{L_1, \dots, L_n\}$  for  $L_i = \sup_V \|\nabla F_i\|$ .

$\text{PC}^r$  functions possess a first-order approximation called the Bouligand derivative, analogous to the push-forward of a smooth function  $Tf : TD \rightarrow T\mathbb{R}^n$ ,  $Tf(x, v) = (f(x), f'(x; v))$ .

**Definition III.1.** A function  $f$  is called Bouligand differentiable at  $x_0$  if it is directionally differentiable, i.e, the limit

$$f'(x_0; y) = \lim_{\substack{a \rightarrow 0 \\ a > 0}} \frac{f(x_0 + ay) - f(x_0)}{a}$$

exists, and the map  $f'$  is a first order approximation of  $f$ , i.e.

$$\lim_{x \rightarrow x_0} \frac{\|f(x) - f(x_0) - f'(x_0; x - x_0)\|}{\|x - x_0\|} = 0$$

A function  $f$  is called Bouligand differentiable if it is Bouligand differentiable at all  $x$  in its domain. The alert reader will observe that the Bouligand derivative (B-derivative) is positively homogeneous to the first order, but need not be linear. The classical Fréchet derivative is recovered by definition if the map  $f'$  is required to be linear. The B-derivative extends properties of calculus on smooth functions, namely the chain rule [Scholtes \(2012, Theorem 3.1.1\)](#) and the fundamental theorem of calculus ([Scholtes, 2012, Proposition 3.1.1](#)). Moreover,  $PC^r$  functions enjoy a special structure to their B-derivative ([Scholtes, 2012, Prop. 4.1.3](#)), namely that it itself is a  $PC^{(r-1)}$  function given by  $f'(x_0; y) = \{\nabla f_i(x_0) \cdot y \mid i \in I_f^e(x_0)\}$ .  $I_f^e(x_0) := \{i \in \{1, \dots, k\} \mid x_0 \in \text{Cl}(\text{Int}(\{x \in U \mid f(x) = f_i(x)\}))\}$ . In this case, the B-derivative looks like a collection of finitely many pieces of the regular F-derivatives of the selection functions, where in a given direction, the B-derivative is determined by the F-derivative of the active function in that domain.

**Remark.** *Since the B-derivative of a  $PC^r$  function is itself  $PC^{r-1}$ , for suitable  $r$ , higher order B-derivatives are defined analogously, assuming the appropriate limit exists.*

### 3.2.1 Event Selected Vector Fields

The formal definition of an event selected vector field is incumbent upon the notion of an *event function*.

**Definition III.2** ([Burden et al. \(2016\)](#), Def 1). Given a vector field  $F : D \rightarrow \mathbf{T}D$  over an open domain  $D \subset \mathbb{R}^d$ , and a smooth submersion  $h \in C^r(D, \mathbb{R})$ , we say that  $h$  is an *event function* for  $F$  on  $D$  if  $\exists f > 0$  s.t.  $\forall x \in D$ ,  $Dh(x)F(x) \geq f$ . A codimension

one embedded submanifold  $\Sigma$  for which  $h|_{\Sigma}$  is constant is referred to as a local section for  $F$

Note that the requirement of  $h$  to be a submersion is inevitable, as necessarily  $Dh(x) \neq 0$  via the transversality being imposed. Similarly, fixed points of  $F$  are prohibited.

**Definition III.3** ([Burden et al. \(2016\)](#), Def 2). Given a vector field  $F : D \rightarrow \mathbf{T}D$  over an open domain  $D \subset \mathbb{R}^d$ , we say that  $F$  is *event-selected*  $C^r$  at  $\rho \in D$  if there is a point  $\rho \in D$  and a collection of event functions  $\{h_j\}_{j=1}^n$  such that

- (event functions)  $h_j$  is an event function for  $F$  on  $D$  for all  $j \in \{1, \dots, n\}$
- ( $C^r$  extension) for all  $b_n \in \{-1, 1\}^n := B_n$ , with

$$D_b = \{x \in D : (h_j(x) - h_j(\rho))b_j \geq 0\}$$

$F|_{\text{Int}D_b}$  admits a  $C^r$  extension  $F_b : D \rightarrow \mathbf{T}D$  such that  $DF_b(x)h_j(x) \geq v$ , for a constant  $v \in \mathbb{R}_{>0}$

We use the notation  $EC^r$  to denote a vector field that is event-selected at some point  $\rho$ . The second condition is assuming (without a loss of generality) that the zero level sets of the event functions  $h_j$  is where all non-smoothness of  $F$  is confined. Equivalently, the zero level sets of the event functions are the guards of the hybrid system. Note that while the vector field  $F$  must be simultaneously transverse to the collection of event functions, the level sets of  $h_j$  do *not* need to be transverse to each other, which is more general than the prior work reference above ([Dieci and Lopez, 2011](#)), ([Ivanov, 1998](#)).

A essential notion captured by the definition of event-selected systems is the requirement that a minimal amount of progress is made on the value of the event functions  $h_j$ , mandating that arrival at an event surface *must* occur, and that each



transition manifold is crossed exactly once by a given execution. At an intuitive level, the vector field must “climb” the landscape determined by the event functions, e.g., the solutions are always traveling uphill.

We summarize the existence and uniqueness properties in the two following theorems.

**Theorem III.4** (Local Flow: [Burden et al. \(2016\)](#), Theorem 4.). *Suppose the vector field  $F : D \rightarrow TD$  is event-selected  $C^r$  at  $\rho \in D$ . Then there exists a flow  $\phi : \mathcal{F} \rightarrow D$  for  $F$  over a flow domain  $\mathcal{F} \subset \mathbb{R} \times D$  containing  $(0, \rho)$  such that  $\phi \in PC^r(\mathcal{F}, D)$  and*

$$\forall (t, x) \in \mathcal{F} : \phi(t, x) = x + \int_0^t F(\phi(s, x)) \, ds \quad (3.1)$$

While the full proof of the theorem need not be reproduced here, we will provide a sketch that motivates how  $EC^r$  vector fields are natural extensions of smooth ones.

By the definition of an event function,  $h_j(\rho) = 0$ , in which case the event surfaces partition a neighborhood  $U$  of  $\rho$  into  $2^n$  components  $D_b$ ,  $b \in \{-1, 1\}^n$ , where each region is identified by the sign of the event functions  $\{h_j\}_{j=1}^n$ , e.g.,  $b = (-1, 1, 1)$  identifies the region where  $h_1 < 0$ , while  $h_2, h_3 > 0$ . Since the vector  $F$  has its discontinuities restricted to the event surfaces,  $F_b|_{\text{Int}(D_b)} = F|_{\text{Int}(D_b)}$ . Each  $F_b$  is a smooth vector field defined over all of  $D$ , so via classical results, each has a maximal smooth flow  $\varphi_b : \mathcal{F}^b \rightarrow U$ ,  $\mathcal{F}^b \subset \mathbb{R} \times U$ , for  $U \subset D$ . As the smooth extensions  $F_b$  also have the surfaces determined by  $\{h_j\}_{j=1}^n$  as transverse local sections, the implicit function theorem produces locally defined time-to-impact maps  $\tau_b^{h_j} : U_b^{h_j} \rightarrow h_j^{-1}(0)$ .  $U_b^{h_j}$  is an open set containing  $\rho$ , such that,

$$\forall x \in U_b^{h_j} : \left( \tau_b^{h_j}(x), x \right) \in \mathcal{F}_b, \text{ and } \varphi_b(\tau_b^{h_j}(x), x) \in h_j^{-1}(0)$$

Repeated for each extension, this produces a collection of smooth flows  $\varphi_b$  and impact time maps  $\tau_b$  jointly defined on some open set  $V$  containing  $\rho$ . The contents of

the proof of Thm III.4 is demonstrating that the flow of the vector field  $F$  can be constructed via composing these smooth objects together, so that the resultant  $\text{PC}^r$  function has smooth pieces determined by  $\phi_b$ , with non-smooth points at the event surfaces.

By Eqn. 18 of [Burden et al. \(2016\)](#), there exist functions  $\tau_b^+ : \mathbb{R} \times D \rightarrow \mathbb{R}$  and  $\tau_b^- : \mathbb{R} \times D \rightarrow \mathbb{R}$  that are the “budgeted time-to-impact” functions. Each returns the positive or negative time, respectively, required to flow out of a region  $D_b$  without exceeding a given time budget. With these budgeted times, loosely speaking, the  $\text{PC}^r$  functions  $\varphi_b^+(t, x) = (t - \tau_b^+(t, x), \phi_b(\tau_b^+(t, x), x))$  and  $\varphi_b^-(t, x) = (t - \tau_b^-(t, x), \phi_b(\tau_b^-(t, x), x))$  are defined. Each flows a region  $D_b$  forward or backward in time until the trajectory  $\phi_b$  crosses the closest event surface, yet for times larger (respectively smaller) than the arrival time, it is the identity map. The local flow can then be defined as ([Burden et al., 2016](#), eq.30)

$$\phi = \pi_2 \circ \left( \prod_{b=-\mathbb{1}}^{+\mathbb{1}} \varphi_b^+ \right) \circ \left( \prod_{b=+\mathbb{1}}^{-\mathbb{1}} \varphi_b^- \right) \quad (3.2)$$

Every  $\varphi_b$  is jointly defined on some open neighborhood of  $\rho$ , so that by composing *every*  $\varphi_b$  in lexicographic sequence (though, any causal order will suffice) determined by  $B_n$ , a flow is defined that coincides with the flow in  $F_b$  when  $F_b$  is active. For details, see [Burden et al. \(2016\)](#).

With this strategy, it is clear that the smooth extension chosen in Def III.3 does not effect the outcome, as the associated  $\varphi_b$  is stationary in the extended domain.

**Corollary III.5** (Global Flow: [Burden et al. \(2016\)](#), Corollary 5). *If  $F$  is  $\text{EC}^r$  at  $\rho \in D$ , then there exists a unique maximal flow  $\phi \in \text{PC}^r(\mathcal{F}, D)$  for  $F$ . This flow has the following properties:*

1. *For each  $x \in D$ , the curve  $\phi^x : \mathcal{F}^x \rightarrow D$  is the unique maximal integral curve of  $F$  starting at  $x$ .*
2. *If  $s \in \mathcal{F}^x$ , then  $\mathcal{F}^{\phi(s, x)} = \mathcal{F}^x - s = t - s : t \in \mathcal{F}^x$*

3. For each  $t \in \mathbb{R}$ , the set  $D_t = \{x \in D : (t, x) \in \mathcal{F}\}$  is open in  $D$  and  $\phi_t : D_t \rightarrow D_{-t}$  is a piecewise- $C^r$  homeomorphism with inverse  $\phi_{-t}$ .

Earlier, we observed that others previously established the existence and continuity of the flow for more general classes of discontinuous vector fields. Here, the flow of an  $EC^r$  systems is shown to be  $PC^r$ , a more structured set of functions compared to the set of merely continuous functions, especially in light of piecewise-differentiable calculus. We remark that the flow  $\phi$  is Lipschitz continuous due to being  $PC^r$ .

Given that a key tool used in the construction of the global flow is the local section property of the event functions, which is an open condition, it is reasonable to expect persistence and continuity of the flow under structural perturbations to either the vector field  $F$  or its associated event functions. Indeed, that is the case, as summarized in the following two results, prefaced with some notation. The space  $C^r(\amalg_{b \in B_n} D, \amalg_{b \in B_n} TD)$  is regarded as a vector space under pointwise addition of tangent vectors, and given the norm

$$\|F\|_{C^r} := \sum_{b \in B_n} \left\| F|_{\{b\} \times U} \right\|_{C^r}$$

That is, it takes the  $C^r$ -norm of each selection function when it is active, and sums them up. The norm is taken to be uniform, in that  $\|F - G\|_{C^r} < \epsilon \implies \forall x \in U, \forall k \leq r, \|(D^k F - D^k G)\|_{op} < \epsilon$ .

**Theorem III.6** (Vector Field Perturbation, [Burden et al. \(2016\)](#), Theorem 12). *Let  $F \in C^r(\amalg_{b \in B_n} D, \amalg_{b \in B_n} TD)$ ,  $h \in C^r(D, \mathbb{R}^n)$  determine an  $EC^r$  vector field at  $\rho \in D, r \geq 1$ . Then for all  $\epsilon > 0$  there exists  $\delta > 0$  such that for all  $\tilde{F} \in B_\delta^{C^r}(F)$ .*

1. Pairing  $h$  with  $\tilde{F}$  determines an  $EC^r$  vector field at  $\rho$
2. The perturbed flow  $\tilde{\phi} : \tilde{\mathcal{F}} \rightarrow D$  obtained by III.4 to  $\tilde{F}$  satisfies  $\tilde{\phi} \in B_\delta^{C^0}(\phi)$  on  $\mathcal{F} \cap \tilde{\mathcal{F}}$  and  $(0, \rho) \in \mathcal{F} \cap \tilde{\mathcal{F}}$

3. There exists a piecewise-differentiable homeomorphism  $\eta \in \text{PC}^r(U, \tilde{U})$  defined between neighborhoods  $U, \tilde{U} \subset D$  of  $\rho$  such that  $\eta|_{B_{\delta(\rho)}} \in B_\epsilon^{C^0}(\text{id}_{B_{\delta(\rho)}})$  and

$$\eta \circ \phi(t, x) = \tilde{\phi}(t, \eta(x)) \quad (3.3)$$

For all  $(t, x) \in \mathbb{R} \times \mathbb{R}^d$  such that  $x \in U, t \in \mathcal{F}^x \cap \tilde{\mathcal{F}}^{\eta(x)}$ , and  $\phi(t, x) \in U$ .

We note that the stipulation that  $r \geq 1$  is essential, as otherwise, the collection  $\{h_j\}_{j=1}^n$  may not be event functions for the perturbed vector field  $\tilde{F}$ .

**Theorem III.7** (Event Function Perturbation, [Burden et al. \(2016\)](#), Theorem 13). *et  $F \in C^r(\Pi_{b \in B_n} D, \Pi_{b \in B_n} TD)$ ,  $h \in C^r(D, \mathbb{R}^n)$  determine an  $EC^r$  vector field at  $\rho \in D, r \geq 1$ , and suppose that  $Dh(\rho)$  is invertible. Then,  $\forall \epsilon > 0$  sufficiently small there exists a  $\delta > 0$  such that for all  $\tilde{F} \in B_\delta^{C^r}(F), \tilde{h} \in B_\delta^{C^r}(h)$ :*

1. There exists a unique  $\tilde{\rho} \in B_\delta(\rho)$  such that  $\tilde{h}(\tilde{\rho})$  and  $\tilde{h}(x) \neq 0$  for all  $x \in B_\delta(\rho) \setminus \{\tilde{\rho}\}$ ;
2. Pairing  $\tilde{h}$  with  $\tilde{F}$  determines an event-selected  $C^r$  vector field at  $\rho$ ;
3. The perturbed flow yielded by theorem III.4,  $\tilde{\phi} : \tilde{\mathcal{F}} \rightarrow D$  satisfies  $\tilde{\phi} \in B_\epsilon^{C^0}(\phi)$  on  $\tilde{\mathcal{F}} \cap \mathcal{F}$ .
4. There exists a piecewise-differentiable homeomorphism  $\eta \in \text{PC}^r(U, \tilde{U})$  defined between neighborhoods  $U, \tilde{U} \subset D$  containing  $\{\rho, \tilde{\rho}\}$  such that  $\eta|_{B_{\delta(\rho)}} \in B_\epsilon^{C^0}(\text{id}_{B_{\delta(\rho)}})$  and

$$\eta \circ \phi(t, x) = \tilde{\phi}(t, \eta(x)) \quad (3.4)$$

For all  $(t, x) \in \mathbb{R} \times \mathbb{R}^d$  such that  $x \in U, t \in \mathcal{F}^x \cap \tilde{\mathcal{F}}^{\eta(x)}$ , and  $\phi(t, x) \in U$

The above perturbation results show that the solutions of an  $EC^r$  system are continuous with respect to perturbation of their determining parameters. It is important that such a property hold, since we hope to apply event-selected theory to physical

systems – if they did not, the theory would be too fragile to use on real objects with noise and finite precision.

The perturbation of event functions preserving the topological structure of the flow suggests appealing relevance for legged locomotion. If crossing of event surfaces coincides with limb contact or removal from the ground, the results in III.6 and III.7 indicate that transitions that happen early or late (within some neighborhood of nominal) will not cause a catastrophic change in response of the robot. In the sequel we will employ this quality to argue that multi-legged robots described via event-selected vector fields incorporate automatic resiliency against uneven ground and realistic implementation constraints (finite bandwidth, processing time, etc).

### 3.3 First Order Properties of Event Selected Systems

The above theorems establish that event selected systems admit solutions and are topologically tolerant of perturbation. However, it still represents a class of objects difficult to analyze. Even conventional nonlinear systems are rife with challenges in their analysis, but we eventually hope to establish not just foundational properties of  $EC^r$  behavior, and also articulate their utility for the modeling and control of legged robots. One of the most prolific tools for the stability analysis of smooth nonlinear dynamical systems is differential linearization, wherein a first order variational equation that is more tractable to analyze is used over the original dynamics to estimate local behavior along a trajectory.

The first order behavior of a smooth flow around a particular solution curve is determined with using the variational equation. It is a classic result ([Hirsch et al., 2012](#), pg. 151) that for  $F \in C^r(\mathbb{R}^n, \mathbb{R}^n)$ ,  $r \geq 1$ , the variational equation is given by

$$\frac{d}{dt}X(u) = D_x F|_{\phi(u,x)} \cdot X(u), \quad X(0) = X_0 \quad (3.5)$$

The key feature deduced the variational equation is that  $D_x\phi(t, x) = X(t)$  (or conversely, defines it), which allows us to approximate the true nonlinear solutions for initial conditions near  $x(0)$  to first order as  $\phi(t, x(0)) + X(t)$  by integrating Eqn. (3.5). We also remark that  $X(t) \cdot F(x_0) = F(\phi(t, x_0))$  as  $D_x\phi(t, x) \cdot F(x) = F(\phi(t, x))$ .

For an  $\text{PC}^r$  flow, a similar first-order approximation holds, but the associated variational equations must account for the event surfaces. In the case of a single surface with either transversal crossing or sliding modes, previous authors ([Müller, 1995](#); [Dieci and Lopez, 2011](#)) established that the variational equation must be updated discontinuously with the so-called *saltation matrix* which accounts for the differing arrival times of solutions starting nearby the trajectory of linearization. Extensions to multiple intersecting surfaces of discontinuity were considered in [Ivanov \(1998\)](#), [Bernardo et al. \(2008\)](#), [Dieci and Lopez \(2011\)](#), and [Bizzarri et al. \(2013\)](#), where each surface crossing induces multiplication by a saltation matrix. As such, the computation of the first-order approximation of the flow of an event selected vector field using this approach has combinatorial complexity. Due to this, the cited authors (e.g., [Ivanov \(1998\)](#)) establish that the computation of the saltation matrix for a number of surfaces larger than two is possible, but not provide explicit expressions. In an algorithmic improvement, the results continue from [Burden et al. \(2016\)](#) to synthesize the full variational equation with jump discontinuities using  $\text{PC}^r$  calculus, where the combinatorial complexity of the transition order is encoded in the Bouligand derivative of the flow, whose definition, when evaluated, will capture all cases. Since the full statement [Burden et al. \(2016\)](#) contains inordinate notational preliminaries, we will omit a full reproduction and summarize an important insight through an illuminating special case. The above definition of an  $\text{EC}^r$  vector field implies that for any  $x \in D_{-1}$ , given a large enough time  $t_f$ , will satisfy  $\phi(t_f, x) \in D_{+1}$ . To illustrate the worst case (in that the largest possible update of saltation matrices need to be determined), further more assume that  $\phi(s, x) = \rho$  for some time  $s$ , i.e. the number

of saltation matrices differing from the identity is maximized. By taking the Bouligand derivative of Eqn. 3.2, which satisfies the chain rule, we may immediately write (avoiding the details of the indexing of  $j$ , as much of the aforementioned notation is making it precise, but it accounts for each flow domain other than the last),

$$D\phi(t, x : v, w) = D\phi(t - s, \rho) \circ \left[ \prod_{j=1}^m D\varphi_j^+(0, \rho) \right] \begin{bmatrix} 0 \\ D\phi(s, x) \end{bmatrix} \begin{bmatrix} v \\ w \end{bmatrix} \quad (3.6)$$

Where the first and last terms expressions in  $D\phi$  are restricted to  $b = -1$  and  $b = +1$ . In those regions,  $\phi$  is classically differentiable, so that  $D\phi$  arises from the classic variational equation, Eqn. (3.5). It is immediate that the saltation matrices are exactly the term  $\left[ \prod_{j=1}^m \varphi_j^+(0, \rho) \right]$ , which relates the smooth fundamental matrix pre-transition to the fundamental matrix post-transition. In other words, the saltation matrices provide a discrete update to the variational equation as it encounters a hybrid event. Rather than proceed as authors did in [Ivanov \(1998\)](#), where each surface was considered independently, the usage of  $PC^r$  calculus allows the construction of the saltation matrices in direct terms of the flow and its component vectors fields.

The conclusion of this discussion of the first-order properties of  $EC^r$  systems it that they admit much of the same kind of first-order approximation that the smooth case does. Additionally, first order properties are derived using  $PC^r$  calculus, which generalizes classic calculus in familiar ways. As a result, local behavior, stability, and other questions determined via linearization for smooth systems can be analogously extend to  $EC^r$  systems with adjustment. A result of significant relevance is the following corollary.

**Corollary III.8** ([Burden et al. \(2016\)](#), Prop. 14). *Let  $\gamma : \mathbb{R} \rightarrow D$  is a periodic orbit of  $F \in EC^r(D)$ . Let  $P \in PC^r(S, \Sigma)$  be the Poincaré map (exists via ([Burden et al., 2016](#), Theorem 10)) over local section  $\Sigma$ , where  $S \subset \Sigma$  has a fixed point  $P(\rho) = \rho$ , and  $DP$  is a contraction over tangent vectors near  $\rho$ ; i.e. - there exists  $c \in (0, 1)$ ,*

$\delta > 0$ , and  $\|\cdot\| : \mathbb{R}^{d-1} \times \mathbb{R}^{d-1} \rightarrow \mathbb{R}$  such that

$$\forall x \in B_\delta(\rho) \subset S \cap \Sigma, v \in T_x \Sigma : \|DP(x; v)\| \leq c \|v\|$$

Then  $\gamma$  is an exponentially stable periodic orbit.

Cor. III.8 is of supreme relevance for much of the sequel, as we are ultimately interested in employing piecewise-constant event selected vector fields as controllers to synthesize high-performance gaits for a legged robot. As we are implicitly assuming that a gait is a stable periodic orbit, understanding its stability in terms of the Bouligand derivative of the Poincaré map is useful for predicting the the gait's asymptotic properties.

### 3.4 Summary of $EC^r$ Systems

$EC^r$  systems are piecewise-differentiable systems with a finite number of transition manifolds, represented as zero level-sets for the *event functions*  $\{h_j\}_{j=1}^m$ . The event functions locally partition the state space in a neighborhood of a point  $\rho$  into up to  $2^m$  pieces. On each piece, smooth functions  $f_b$ ,  $b \in \mathcal{B}^m := \{-1, 1\}^m$  define dynamics under which individual transitions are irrevocable. Solutions evolving under such dynamics can experience up to  $m!$  different transition sequences when passing through that region of state-space. [Burden et al. \(2016\)](#) show that  $EC^r$  vector fields have a piecewise differentiable and continuous flow ([Burden et al., 2016](#), Cor. 5 ) that is conjugate to a flowbox via a piecewise-differentiable map ([Burden et al., 2016](#), Thm. 11 ). The calculus-like properties of  $PC^r$  functions allow immediate consideration of first-order properties of solutions, which is informative for stability analysis of periodic solutions.



### 3.5 Numerical Estimation of the Bouligand Derivative

While the Bouligand derivative of the flow of an  $EC^r$  vector field may in general be exactly determined by the variational equation, explicit numerical methods are required if we hope to apply the theory of event-selected systems to physical systems. As we saw above, a substantial advantage of  $EC^r$  theory is that the resultant flow is  $PC^r$ , and that  $PC^r$  extends traditional calculus in familiar ways. However, given the relative obscurity of  $PC^r$  calculus, there is a lack of numerical tools that have allowed classical calculus to have such rich application to physical processes, e.g. numerical integration and differentiation.

Much akin to the numerical differentiation tools that use finite step sizes to approximate the Jacobian of a classically differentiable function, a finite step size numerical method can be used to estimate the Bouligand differential  $D\phi$  of the flow  $\phi$  arising from an  $EC^r$  system.

We now present an algorithm to compute the B-derivative at the point of mutual intersection of the event surfaces (we always denote by  $\rho$ ). We recall that  $D\phi$  is a piecewise linear map. Equivalently, we are interested in determining the saltation matrices from Eqn. (3.6); since we can always pre- and post multiply by smooth monodromy matrices for flows of arbitrary length that start in  $D_{-1}$ , and terminate in  $D_{+1}$ , the saltation matrices at  $(0, \rho)$  are the dominant object to compute.

A major difficulty that we must account for is the transition order of an solution, as can be seen in Eqn. (3.6). The transition order of points  $v + \delta$  and  $v - \delta$  in some neighborhood of  $v$  may be different for all  $\delta > 0$ , resulting in  $m!$  linear maps that determine  $D\phi$  around  $\rho$ , which compose into the piecewise-linear B-derivative.

#### 3.5.1 Factorial Complexity of $EC^r$ flows

We now assume that we have a dynamical system with dimension  $d$ , with  $0 \leq m \leq d$  (potentially intersecting) transition manifolds that we wish to numerically

investigate. Unlike the more general case above, we assume that the event manifolds are transverse to each other. Since the flow is smooth in each region that undergoes a fixed transition sequence, we could use conventional numerical differentiation techniques which evaluate at least  $d + 1$  points for each of the  $m!$  transition sequences. Typically, integrating from an initial condition is the obvious way to determine its consequent transition sequence. Therefore, even assuming the results are cached, for  $n$  data points one would need  $m!(d + 1) + n$  integrations. We present an algorithm that instead requires  $2^m - m + d + n$  integrations, producing a combinatorial  $m!/2^m$  improvement in execution time. If  $m \gg 1$  or  $d \gg 1$ , this is particularly prominent. Even in a simple case, such as the  $d = 12$  dimensional phase space of a single rigid body, with  $m = 3$  contacts, the naive approach requires  $78 + n$  integrations vs. our  $17 + n$  integrations – a better than  $\times 4$  improvement. Moreover, we will do so without ever integrating the non-linear dynamics, a computationally expensive task, which in practice will yield a speed-up far more considerable than that suggested by improvement in the number of integrations needed.

Our algorithm provides this improvement by estimating a first order approximation of the flow via triangulation of state space into  $m!$  regions using  $2^m - m + d$  triangulation points. This triangulation is selected so that all trajectories starting in a given region undergo an identical transition sequence and therefore share the same linearized approximation to the flow. The images of the  $2^m$  triangulation points allow all such linearizations to be recovered efficiently, providing the speedup.

To validate the algorithm, we tested it on cases where the first order approximation is known exactly, as well as on a motivating multi-contact example of a legged chair impacting a springy ground. The multi-legged chair was chosen as it is reminiscent of multi-legged robot and animal models with dynamic limbs, as well as having an obvious attracting equilibrium state — stationary with all legs on the ground — despite the contact sequencing possibilities.

### 3.5.2 Notational Preliminaries

Let  $F : U \rightarrow \mathbb{R}^d$  be an  $EC^r$  vector field on an open set  $U \subseteq \mathbb{R}^d$ ,  $\rho \in U$ , and  $\{h_j : U \rightarrow \mathbb{R}\}_{j=1}^m$  be a finite number of smooth functions. We will define a vector valued transition function  $h(x) := (h_1(x), \dots, h_m(x))$ . We will use the notation  $H_j := h_j^{-1}(h(\rho))$  for the event surfaces, each of which by definition is a codimension 1 embedded submanifold of  $U$ . Since we will refer to several  $EC^r$  systems simultaneously in the sequel, we will use the tuple  $(A, f)$  to refer to vector field  $A$  and transition functions  $f$  that define an  $EC^r$  system.

A key result we concluded in Thm III.4 is that event selected systems have flows that are  $PC^r$ , imbuing them with many useful properties ([Scholtes, 2012](#)). We denote by  $\phi(t, x)$  the result of flowing initial state  $x$  for  $t$  units of time. We additionally employ Thm. III.4 which gives us the property that  $EC^r$  systems admit  $PC^r$  time-to-impact functions for the transition manifolds  $H_j$ . Recall, the time-to-impact function  $\tau_j^F$  satisfies  $\phi(\tau_j^F(x), x) \in H_j$  and is also  $PC^r$ . We refer to the image of  $\tau_F := (\tau_1^F, \dots, \tau_n^F)$  as *impact-time coordinates*, and note that if  $m < d$  then  $h$  can easily be extended to make  $m = d$  and render  $\tau$  invertible (see Eqn. 3.11).

To avoid repetition, we will use  $x \in U$  for a typical point,  $y = h(x) - h(\rho)$  for its image under  $h$  which is in the neighborhood of 0, and  $t = \tau^F(x)$  for its vector of impact times which is also in the neighborhood of 0. For later use, we note that the transform of the vector field into impact-time coordinates is the vector field of all negative ones, i.e.  $-\mathbf{1} := (-1, \dots, -1) \in \mathbb{R}^d$ , because by definition  $\tau \circ \phi(t, x) = -\mathbf{1}t + \tau^F(x)$  ([Burden et al., 2016](#)). The function  $\text{sign} : \mathbb{R}^d \rightarrow B_d$  is the vectorized signum function given by

$$\forall x \in \mathbb{R}^d : j \in \{1, \dots, d\} : e_j^T \text{sign}(x) = \begin{cases} -1, & x_j < 0 \\ +1, & x_j \geq 0 \end{cases} \quad (3.7)$$

We wish to compute the first-order effect of a perturbation  $\delta x$  on an initial con-

dition  $x(0)$  that starts prior to any contact, and evolves to state  $x(t)$  that is post all contacts, i.e., when  $x(0) \in D_{-1}$  and  $\phi(t, x(0)) \in D_{+1}$ . Thus, the B-derivative  $D\phi(t, x; u, v)$ , for  $t$  a sufficiently large value, is the first-order map we are looking for, as  $\phi(t + \delta t, x + \delta x) \approx \phi(t, x) + D\phi(t, x; \delta t, \delta x)$ .

Unlike the classical (Fréchet) derivative, or Jacobian, the B-derivative of the flow is in general not linear, but for  $\text{PC}^r$  functions it is always continuous and *piece-wise* linear ([Scholtes, 2012](#)). We now will use a critical feature of  $\text{PC}^r$  flows, which is that the linear pieces depend solely on the order in which transition manifolds are crossed (as can be seen from Eqn. (3.2)).

### 3.5.3 Impact time order induces a cone decomposition

Define the map  $s : \mathbb{R}^m \rightarrow S_m$  to identify a permutation  $s(t) := \sigma \in S_m$  which sorts the coordinate components of  $t$ , i.e.  $t_{\sigma_1} \leq t_{\sigma_2} \leq \dots \leq t_{\sigma_m}$ . Here  $S_m$  is the *symmetric group of order  $m$*  – the collection of invertible maps from  $\{1, \dots, m\}$  to itself.

Note that this does not define  $s$  uniquely for any  $t \in \mathbb{R}^m$ , because if the times  $t_i$  are not distinct,  $s$  needs only pick some arbitrary permutation among the permutations that sort the  $t$ . For example, if  $t_1 = t_2 < t_3 < \dots$  then both  $(1, 2, 3, \dots)$  and  $(2, 1, 3, \dots)$  are valid values for  $s$ , and either one could be selected arbitrarily. Thus,  $s$  is uniquely defined on a dense open set consisting of points with distinct impact times. For  $t$  with  $t_i$  all distinct, we can define subsets

$$\mathcal{K}_\sigma := \{t \in \mathbb{R}^m \mid s(t) = \sigma, t_{\sigma_i} < t_{\sigma_{i+1}}\}$$

**The  $\mathcal{K}_\sigma$  sets are open** If  $t_i$  are distinct then adding a  $\delta \in \mathbb{R}^d$  such that  $2\|\delta\|_\infty < \min_k(t_{\sigma_{k+1}} - t_{\sigma_k})$  leaves the order unchanged.

**Each  $\mathcal{K}_\sigma$  is a cone, and is unbounded along the direction  $\mathbf{1}$**  For  $\alpha > 0$ ,  $s(t) = s(\alpha t) = s(\alpha \mathbf{1} + t)$ .

**Each  $\mathcal{K}_\sigma$  is finitely generated** We will use the notation  $\chi_A$  to designate the *indicator vector* of a set  $A \subseteq \{1 \dots n\}$  ( $n$  known from context). This is the vector in  $\{0, 1\}^n$  whose components are 1 for elements of  $A$  and 0 otherwise. Using this notation, we define the *prefix vectors* of  $\sigma$  to be

$$p_k^{(\sigma)} := \chi_{\sigma(\{1 \dots k\})} \quad (3.8)$$

i.e.  $[p_k^{(\sigma)}]_i = 1$  if, and only if, there is a  $j \leq k$  such that  $\sigma_j = i$ .

We will often omit the  $\cdot^{(\sigma)}$  superscript when  $\sigma$  is clear from context. Clearly,  $p_0 = 0$ ,  $p_m = 1$ , and if e.g.,  $\sigma_1 = 3$ ,  $\sigma_2 = 1$ , then  $p_1 = (0, 0, 1, 0 \dots)$ , and  $p_2 = (1, 0, 1, 0 \dots)$ . Note now that  $\mathcal{K}_\sigma$  is the interior of the cone span of  $p_0, \dots, p_k$ , i.e.  $\mathcal{K}_\sigma = \text{int}(\{\sum_{k=0}^m \alpha_k p_k \mid \alpha_k \geq 0\})$ .

**The  $\text{cl}(\mathcal{K}_\sigma)$  partition  $\mathbb{R}^m$**  Every point  $x \in \mathbb{R}^n$  is the limit of a sequence of points with distinct coordinates, therefore every point in  $\mathbb{R}^d$  belongs to one or more  $\text{cl}(\mathcal{K}_\sigma)$ .

Thus we have shown that  $\mathcal{K}_\sigma$  is a convex finitely generated open cone, and that the collection of all such closed cones covers  $\mathbb{R}^m$ . Note that if we had used the convex hull operation of the prefix vectors instead of a cone span operation, we would have obtained the venerable and closely related Kuhn decomposition of the hypercube into simplices ([Kuhn, 1960](#)). The interested reader should consult [Scholtes \(2012, Prop. 2.2.6\)](#) for further details and proofs concerning the decomposition of a piecewise-linear map over conical domains.

### 3.5.4 Definition of the sorting map

For an event selected system  $(F, h)$ , we define the *sorting map* of  $(F, h)$  to be the composition  $r_F := s \circ \tau^F$ ,  $r_F : \mathcal{U} \rightarrow S_m$ .  $r_F$  is a map which identifies a permutation that orders the impact times. We suppress explicit inclusion of the dependence of  $\tau^F$  on  $h$  for notational clarity. We will always use unique notation for transition surfaces

in the sequel to remove any ambiguity. It follows from [Burden et al. \(2016, Eqn. 65\)](#) that if  $t$  is large enough so that all transitions have occurred,  $D\phi(t, x; 0, v) = M(r_F(x)) \cdot v$ , where  $M : S_m \rightarrow \mathbf{L}(\mathbb{R}^d, \mathbb{R}^d)$  is a linear operator for each sequence of contacts, i.e. the  $m!$  pieces of the B-derivative of the flow are a collection of matrices indexed via  $S_m$  through the map  $r_F$ . The matrices  $M$  are the conventional Jacobians that arise when the flow is restricted to a domain  $(\tau^F)^{-1}(\text{cl}(\mathcal{K}_\sigma))$  that undergoes a fixed transition sequence, since on each such domain the flow  $\phi(t, x)$  for  $t$  large is smooth.

Take  $m + 1$  affinely independent<sup>1</sup> points  $\{t_i\}_{i=1}^{m+1}$  in the cone  $\mathcal{K}_\sigma$ , and define  $\beta := \max_i \|t_i\|_\infty$ . Because the flow in impact time coordinates is the constant vector  $-\mathbf{1}$ , after  $\beta$  units of time all the  $t_i$  points will have all coordinates non-positive with values  $t_i - \beta\mathbf{1}$ , indicating that their transitions happened in the past. Remapping back to the original coordinates,  $t_i = \tau^F(x_i)$  for some choices of  $x_i$ , and  $t_i - \beta\mathbf{1} = \tau^F(x'_i)$  for the specific  $x'_i = \phi(\beta, \beta x_i)$ . At the limit  $\beta \rightarrow 0^+$ ,  $\phi(\beta, \beta x_i) \rightarrow M(\sigma) \cdot x_i + o(\beta^2)$ , where  $M(\sigma)$  is the unique linear map mapping  $x_i$  to  $x'_i$  for all  $i$ . This map is  $M(\sigma)$  is the “piece” of the B-derivative  $D\phi(\beta, \rho; 0, \cdot)$  corresponding to the fixed transition sequence  $r_F(x_i) = \sigma$ , which is by construction, the same  $\sigma$  for all the  $x_i$  in question.

This computation would not be advantageous for a general non-linear vector field  $F$ , as finding the map  $\tau^F$  at a point is equivalent to integration of the vector field  $F$  from that point, which is generally numerically burdensome. However, in the case of  $\text{EC}^r$  systems we need not integrate  $F$ . The flows of  $\text{EC}^r$  system have a local approximation called the *sampled vector field* ([Burden et al., 2016, §7.1.3](#)), a piecewise constant  $\text{EC}^r$  vector field  $\tilde{F}$ , whose pieces are defined by an arrangement of hyperplanes, and whose flow  $\tilde{\phi}$  matches  $\phi$  to first order at  $(0, \rho)$ , i.e.  $D\tilde{\phi}(0, \rho; u, v) = D\phi(0, \rho; u, v)$ . We

---

<sup>1</sup>points  $v_0, \dots, v_k$  are affinely independent if  $v_1 - v_0, \dots, v_k - v_0$  are linearly independent; equivalent, if any strict subset of the vectors span a strict subset of the affine span. Observe this is not the same as being linearly independent - we only want the “difference vectors” to be linearly independent.

define the following. For all  $b \in \mathcal{B}^n$ , let

$$D_b := \{x \in U \mid b_j \cdot Dh_j(\rho)(x - \rho) > 0\} \quad (3.9)$$

and let  $\bar{D}_B$  be the closure of  $D_b$ . By definition,  $F|_{\bar{D}_B} = f_b|_{\bar{D}_B}$  is a smooth vector field. By taking the limiting value of  $f_b$  at  $\rho$ , we subsequently define the piecewise-constant vector field  $\tilde{F} : U \rightarrow \mathbf{T}U$  by

**Definition III.9** (Sampled Vector field).

$$\forall b \in \mathcal{B}^n, \forall x \in D_b : \tilde{F}(x) := f_b(\rho) \quad (3.10)$$

For all points  $x \in \partial \bar{D}_b$ ,  $\tilde{F}(x) = F(\rho)$ .

$\tilde{F}(x)$  is called the sampled vector field associated to  $F$ , and its flow is denoted  $\tilde{\phi}$ , which is called the *corner flow*.

### 3.5.5 The corner flow

To simplify the problem, we can always define a change of coordinates so the submanifolds  $H_j$  are re-mapped to the standard coordinate planes. Indeed, if  $m = d$ , the function  $h$  is already the required coordinate change. If  $m < d$ , a diffeomorphism can be completed from  $h$ , through the following procedure. Choose a matrix  $Q \in \mathbb{R}^{(d-m) \times d}$  such that  $\text{Im}(Q^T) = \ker(Dh_\rho)$ , i.e.  $\ker(Q) = \ker(D\rho)^\perp$ . Define

$$\hat{h}(x) := (h(x) - h(\rho), Q \cdot (x - \rho)) \quad (3.11)$$

By construction  $D\hat{h} = Dh_\rho \oplus Q$  is invertible and thus  $\hat{h}$  is a local diffeomorphism from a neighborhood of  $V$  of  $\rho$  to a neighborhood of  $0 = \hat{h}(\rho)$ . Let  $W := \hat{h}(U \cap V)$  be the image of the entire domain  $V \cap U$  under this map, and  $\phi^W : \mathbb{R} \times W \rightarrow W$  be the flow  $\phi$  conjugated into these new coordinates  $\phi^W(t, y) := \hat{h}(\phi(t, \hat{h}^{-1}(y)))$ .

It is tempting to try and reduce the  $m < d$  case to the  $m = d$  case by extending the submersion  $h$  to a diffeomorphism into  $\mathbb{R}^d$ . It is always possible to extend the function  $h$  to a local diffeomorphism  $\hat{h}$  such that the *sampled* vector field is event-selected with respect to *every* component of  $\hat{h}$  (note that above, the Lie derivatives of the components defined by  $Q$  is not required to be positive). Whether or not it is possible for the non-linear case is not immediately apparent. See appendix A for details. More importantly we will demonstrate that such a rectification is *unnecessary* and *undesirable*, as it would only act to increase computational complexity.

We now change coordinates through  $\hat{h}$  to simplify the computation. Since we're interested in the sampled vector field, we denote with  $\tilde{G}$  the transform of  $\tilde{F}$  under  $\hat{h}$ .

As we did in the definition of  $\tilde{F}$ ,  $\tilde{G}$  has components  $G_b$ . By construction,

$$\forall j, N_j := \hat{h}(H_j) = \{y \in h(U) \mid \langle e_j, y \rangle = 0\} \quad (3.12)$$

where  $e_i$  denotes the  $i$ -th standard basis vector. Then the new vector field components  $G_b$  in the  $\hat{h}$  coordinates, for a given  $b \in \{-1, 1\}$ , are

$$G_b(y) = D\hat{h} \cdot \tilde{F}(\hat{h}^{-1}(y)) \quad (3.13)$$

The map  $\hat{h}$  also defines the *generalized orthants*

$$\tilde{D}_b := \left\{ y \in \text{Dom } \tilde{G} \mid b \in \mathcal{B}^d, b \cdot y > 0 \right\} \quad (3.14)$$

Now, for  $g_j(y) := \langle e_j, y \rangle$ ,  $j = 1, \dots, m$ , we see clearly, at  $y = \hat{h}(x)$ ,

$$Dg_j(y) \cdot \tilde{G}(y) = [Dh(x) \cdot F(x)]_j = Dh_j(x) \cdot F(x) \geq f. \quad (3.15)$$



Therefore, the general value of  $\tilde{G}$ , which is still piecewise-constant, is given by,

$$\tilde{G}(y) = \tilde{G}_{\text{sign}(y)}(y) = \tilde{G}_{\text{sign}(y)} \quad (3.16)$$

Thus,  $(G, g)$  is event-selected (so has associated maps  $r_G, \tau^G$ , etc.) at  $0 \in W$  w.r.t the first  $m$  coordinates. It consists of  $2^m$  components of dimension  $d$ . Its sampled companion,  $(\tilde{G}, D_0g)$ , has a flow by  $\tilde{\phi}^W : \mathbb{R} \times \mathbf{T}_0W \rightarrow \mathbf{T}_0W$ , and we denote its sorting order map as  $r_G : \mathbf{T}_0W \rightarrow S_m$  (As a matter of notation, we will just use the “ $\sim$ ” to denote objects in a tangent space). The transition manifolds of  $\tilde{G}$  are the first  $m$  coordinate axes, and the uniform lower bound on the rate of crossing those necessarily holds:  $\langle e_i, G_b \rangle \geq f > 0$  for  $i \in \{1, \dots, m\}$ . For the other coordinates, we only conclude  $\langle e_i, G_b \rangle \in \mathbb{R}$  for  $i \in \{m+1, \dots, d\}$  – there is no obvious bound.

If  $m < d$ , the impact time maps do not define a homeomorphism. However, instead let  $\pi_m : \mathbb{R}^d \rightarrow \mathbb{R}^m$  be the standard projection, i.e.  $\pi_m(a, b) = (a, 0), a \in \mathbb{R}^m$ . Then, it is immediately clear that  $r_{\tilde{G}} \circ \pi_m = r_{\tilde{G}}$ , as the map  $r_{\tilde{G}}$  has no dependence on the remaining  $d - m$  coordinates in which no transitions occur. In other words, the flow of  $G$  only experiences vector field discontinuities when its first  $m$  components cross 0, and is otherwise continuous in the remaining coordinates.

$\mathbb{R}^d$  and  $\mathbf{T}_x\mathbb{R}^d$  are isomorphic as vector spaces in an obvious way ([Lee, 2013](#), Prop. 3.2). If  $e_1, \dots, e_d$  is a basis for  $W$ , the basis  $\frac{\partial}{\partial e_1}, \dots, \frac{\partial}{\partial e_n}$  is a natural basis for  $\mathbf{T}_0W$ . In these coordinates, the isomorphism is given by  $y = (y_1, \dots, y_d) \mapsto \delta y = (y_1, \dots, y_d)$ , which we name  $\mathfrak{J}$ . Through  $\mathfrak{J}_U$ , we can similarly identify the sets (or any set)  $\tilde{D}_b$  and  $N_j$  as subsets of  $\mathbf{T}_0W$ . Thus, we deliberately employ the notation “ $y \in \mathbf{T}_0W$ ” to implicitly mean  $\delta y \in \mathbf{T}_0W$  such that  $\mathfrak{J} = \delta y$ , conflating  $W$  and  $\mathbf{T}_0W$ . Analogously, we define  $\mathfrak{J}_U : U \rightarrow \mathbf{T}_\rho U$  to be the coordinate isomorphism on  $U$ . Equivalently, elements from an open neighborhood and tangent vectors can be conflated without ambiguity.

For the sampled vector field, it follows from [Burden et al. \(2016, Eqn. 64\)](#) that the  $\tau^{\tilde{G}}$  is piecewise linear. In fact, for  $\tilde{G}, y \in \mathbf{T}_0 W$ , the components are:

$$\tau_i^{\tilde{G}}(y) := \frac{y_i}{G_b^i}, \quad y \in \tilde{D}_b \quad (3.17)$$

What we can conclude from the proceeding is that an  $\mathbf{EC}^r$  system locally looks like a flow box via iterated “coordinate changes”, diagrammatically as:

$$\begin{array}{ccccc} (F, h) & \xrightarrow{\hat{h}} & (G, g) & \xrightarrow{\tau^G} & (\mathbf{1}, \text{id}) \\ \rho, x \in U & & 0, y \in W & & 0, t \in T \\ \mathcal{I}_U \Big| \lim_{x \rightarrow \rho} & & \mathcal{I}_W \Big| \lim_{y \rightarrow 0} & & \Big| \lim_{t \rightarrow 0} \\ \left( \tilde{F}, Dh \Big|_{\rho} \right) & \xrightarrow{D\hat{h}} & \left( \tilde{G}, Dg \Big|_0 \right) & \xrightarrow{D\tau^G} & (\mathbf{1}, \text{id}) \\ 0, \delta x \in \mathbf{T}_p U & & 0, \delta y \in \mathbf{T}_0 W & & 0, t \in \mathbf{T}_0 T \end{array} \quad (3.18)$$

$F$  is sampled at  $\rho$  to produce the piecewise constant vector field  $\tilde{F}$  taking on the values  $F_b$ .  $\tilde{F}$  is transformed by  $h$  to  $\tilde{G}$ , which is conjugate to a flowbox by the piecewise linear impact time map  $\tau^G$ . The quotes “.” above are in reference to the fact that the map  $\tau^G$  is not a coordinate change for the  $m < d$  case, but rather a projection into  $\mathbb{R}^m$ .

Also implied by the diagram is that the sorting time maps  $\{r_A\}_{A \in \{F, G\}}$  are locally preserved – the transition sequence is unchanged (for an potentially increasingly small open domain) between these transformations. This is obvious for coordinate changes  $h$ . However, it must be shown that the limiting process that yields the sampled vector field also has this property. If the transition order between the non-linear and sampled vector fields are different, we would not expect their B-derivatives to agree.

We will see in the sequel that only the first  $m$ -coordinates are essential to determining the map  $D\phi$ , as they alone determine the transition sequences.

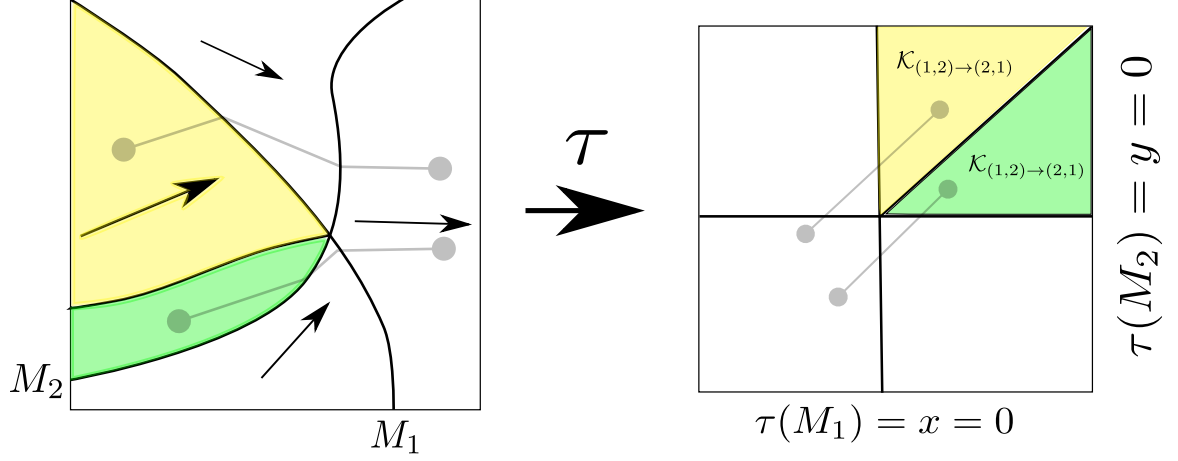


Figure 3.1: The  $PC^r$  impact time map  $\tau$  rectifies a general set of points that share the same transition sequence into a cone in  $\tau$  coordinates. It additionally maps the vector field to a flowbox. The light yellow indicates a subset of points that all have transition sequence  $M_1, M_2$ , while the green domain are points with  $M_2, M_1$  as its transition sequence. The light-grey lines are solution curves, while the dark black lines are transition manifolds.

### 3.5.6 Differential Equality

Recall, we are attempting compute Eqn. (3.6). The principal difficulty in doing so is the appearance of the saltation matrix terms. We conclude that if we can compute  $D\varphi_i^+(0, \rho)$  efficiently, the factorial difficulty of estimating the B-derivative is resolved, as the remaining smooth components are solved classically for large times without requiring a factorial number of integrations. For small times, these monodromy matrices limit to the identity, so can be ignored, and the saltation matrix dominates.

For simplicity, we assume the system has been transformed under  $\hat{h}$  to  $W$ -coordinates. Since  $PC^r$  calculus obeys chain rule, there are no difficulties transforming back to the original  $U$  coordinates. Additionally, we assume employ  $\mathfrak{J}$  implicitly, and assume that  $(G, g)$ , and  $(\tilde{G}, D_0g)$  are defined on the same set  $W$ . We do that as we are interested in the B-derivative as a first-order approximation. A coordinate-free perspective would require either jet bundles or Riemannian geometry, the complexities of both we can avoid since being  $EC^r$  is a local property. Concordantly, we abuse notation, and drop

the “ $W$ ” from  $\varphi$  and  $\tilde{\varphi}$ . Any reference to the “non-linear” or “sampled” system in this section refer to  $(G, g)$  or  $(\tilde{G}, D_0g)$  respectively.

We now state a key fact, which follows from Eqn. 17 of [Burden et al. \(2016\)](#).

$$D\tau(0) \cdot \delta y = D\tilde{\tau}(0) \cdot \delta y \quad (3.19)$$

Eqn. (3.19) follows as the  $D\tau(0)$  depends only on the value of  $G$  at 0, and  $Dg$  at 0. By definition of the sampled vector field, these quantities are the same, so the result follows. It should be intuitively reasonable that the the impact times agree to first order at  $\rho$ , as the nonlinear and sampled vector field flows agree to first order at 0.

We saw above that, for  $\omega : \{1, \dots, m\} \rightarrow \{1, \dots, m\}$  the permutation that indicates the transition order,

$$\Xi_\omega := \prod_{j=1}^m D\varphi_{\omega(j)}^+(0, 0) \quad (3.20)$$

Let  $y \in \tilde{D}_b$  be a curve such that at  $\tilde{s} \in \mathbb{R}_{>0}$ ,  $\tilde{\phi}(\tilde{s}, y) = 0$ ; then, since the sampled vector field is *also* event selected, we obtain an analogous equation,

$$\tilde{\Xi}_{\tilde{\omega}} := \prod_{j=1}^m D\tilde{\varphi}_{\tilde{\omega}(j)}^+(0, 0) \quad (3.21)$$

Wherein again we denote the sampled vector field structures with  $\sim$ , we see immediately from Eqn. (3.19) that,

$$D\varphi_i^+(0, 0) = D\tilde{\varphi}_i^+(0, 0), \quad \forall i = 1, \dots, m \quad (3.22)$$

While this tells us that the individual “pieces” of the sampled vector field and nonlinear B-differential agree, as  $\Xi_\sigma = \tilde{\Xi}_\sigma$ , it does not directly establish that for a given  $\delta x$ , they agree; putatively, the sequences  $\omega$  and  $\tilde{\omega}$  could disagree. Equivalently, we want that if:  $\tau_{\sigma_1} \leq \tau_{\sigma_2} \leq \dots \leq \tau_{\sigma_m}$ , then  $\tilde{\tau}_{\sigma_1} \leq \tilde{\tau}_{\sigma_2} \leq \dots \leq \tilde{\tau}_{\sigma_m}$

Equivalently, we wish to know  $\forall i, j$ , if

$$|\tau_i(x) - \tilde{\tau}_i(x)| \leq |\tau_j(x) - \tau_i(x)| \quad (3.23)$$

Let  $B_r(0)$  be an open ball around the origin, for  $r$  to be determined. By Eqn. (3.19), we know that for  $r$  suitably small,  $x \in B_r(0)$ ,  $\tau_i$  and  $\tilde{\tau}_i$  agree to  $O(\|x\|)$ , i.e., their difference is higher order.

I.e., let  $\Delta\tau_i(x) = \tau_i(x) - \tilde{\tau}_i(x)$ , then,

$$\Delta(x) = o(h) \quad (3.24)$$

Where,  $o(h) \rightarrow 0$ , as  $h \rightarrow 0$ <sup>2</sup>. Assuming that  $D\tau_j(x) \neq D\tau_i(x)$ , the R.H.S of Eqn. (3.23) has *first*-order error. Ergo, if  $r$  is taken suitably small, the difference between the sampled and nonlinear impact times is smaller than the difference between any two components of  $\tau$ , so that the impact order is preserved.

The assumption that  $D\tau_j(x) \neq D\tau_i(x)$  is reasonable, as by Eqn. 17 of [Burden et al. \(2016\)](#), this could only happen if the guard manifolds were not transverse, which violates our assumptions on the rank of  $h$  in this paper.

Thus, on a suitably small neighborhood of the origin, the impact sequence of the nonlinear and sampled vector field agree for  $x$  in the interior of a cone  $\mathcal{K}_\sigma$ .

### 3.5.7 Leveraging differential equality

Our algorithm will compute  $\tilde{\phi}^W(1, \delta x)$  on a restricted domain of  $\tilde{D}_{-1}$ ; the domain will be chosen so that the B-derivative of the flow will be given by Eqn. (3.6). equivalently, the time  $t = 1$  will evolve all states through every transition sequence. Since  $\hat{h}$  is a diffeomorphism, we can transform from  $W$  coordinates to  $U$  coordinates to determine  $\Xi_\omega$ , and thus,  $D_x\phi(t, x)$ .

---


$$^2 f(h) = o(g(h)) \iff \lim_{h \rightarrow 0} \frac{f(h)}{g(h)} = 0$$

Then, we have an efficient process to compute  $D_x\tilde{\phi}(1, \delta x)$  directly. More precisely, let  $\tilde{\phi} : \mathbb{R} \times \mathbb{R}^d \rightarrow \mathbb{R}^d$  be the  $\mathbf{PC}^r$  (corner) flow of  $\mathbf{EC}^r$  (sampled) vector field  $\tilde{F} : \mathbb{R}^d \rightarrow T\mathbb{R}^d$  and  $s, t \in \mathbb{R}$ ,  $x, \rho \in \mathbb{R}^d$  be such that  $0 < s < t$  and  $\rho = \tilde{\phi}(s, x)$ . Given  $\delta x \in T_x\mathbb{R}^d$ , a perturbation away from  $x$  in the  $\delta x$  direction, and letting  $\sigma(\delta x)$  denote the corresponding transition sequence<sup>3</sup> and  $\Xi_{\sigma(\delta x)} \in \mathbb{R}^{(d+1) \times (d+1)}$  the corresponding saltation matrix (*Burden et al., 2016*, Eqn. (67)),

$$D_x\tilde{\phi}(t, x; \delta x) = \begin{bmatrix} F_{+1} & I_d \end{bmatrix} \tilde{\Xi}_{\sigma(\delta x)} \begin{bmatrix} 0_d^\top \\ I_d \end{bmatrix} = M(\sigma(\delta x)). \quad (3.25)$$

Finally, the impact time map  $\tau^G$  for  $G$  is piecewise linear with pieces corresponding to distinct transition sequences, which implies that the conical partition generated by impact times persists in  $y$  coordinates, i.e.  $(\tau^G)^{-1}(\mathcal{K}_\sigma)$  is still a cone in  $y$ , we which will denote  $\mathcal{K}_\sigma^G$ . This follows from the fact that linear maps take cones to cones.

Together these observations imply we can find  $D_x\tilde{\phi}(0, 0; \delta x)$  exactly, by representing its action on the cones  $\mathcal{K}_\sigma^G$ . On each cone, the sampled flow is affine - thus, the linear part of the resulting affine map is the derivative (*Groff et al., 2003*). Representing the flow gives us the desired map  $D\tilde{\phi}$ . The  $\hat{h}$  coordinates defined in Eqn. (3.11) make this computation particularly simple, as we will see.

Finally, to determine the value of  $D\tilde{\phi}(0, \rho)(\delta x)$ , we only need to determine the transition sequence for  $v$ , and evaluate Eqn. (3.25).

### 3.5.8 Computing the B-derivative with prefix vectors

We saw above that the prefix vectors determined by the permutation  $\sigma$  of  $\mathcal{K}_\sigma^G$  are a sequence of  $m + 1$  binary words with increasing support. If we correspond the  $k$ -th place to  $h_k$ , we construct a sequence indicating the order in which the transition

---

<sup>3</sup>unique ae if surfaces are transverse

manifolds  $H_j$  are crossed, producing these  $m+1$  binary words. Since we restricted our interest to initial conditions that begin with no transitions having occurred, we may always initialize the binary sequence with  $(0, \dots, 0)$ , where a “0” denotes a crossing that has not occurred, while a “1” indicates one that has. E.g, if a trajectory with initial condition  $y(0)$  crossed  $H_2$ , then  $H_3$ , and finally  $H_1$ , the sequence would be

$$p_0 = (0, 0, 0)$$

$$p_1 = (0, 1, 0)$$

$$p_2 = (0, 1, 1)$$

$$p_3 = (1, 1, 1)$$

Equivalently,  $r_{\tilde{G}}(y(0)) = ((1, 2, 3) \mapsto (3, 2, 1)) \in S_3$ . As we saw in §3.5.4, each prefix vector  $p_k^\sigma$  has an associated point  $q_k^\sigma := \tau(x'_k) = t_k - \beta \mathbf{1}$ ; in  $y$ -coordinates, these are points  $y_k := (\tau^G)^{-1}(p_k)$ <sup>4</sup> that simultaneously intersect the transition manifolds identified by  $p_k$  at  $t = 1$  (see the diagram in (3.18) for a visual aid). We established that  $\mathcal{K}_\sigma$  is spanned via the prefix vectors defined via  $\sigma$ . Since there are  $2^m$  prefix vectors used in the  $m!$  increasing sequences, only  $2^m$  points are required to span all the  $\mathcal{K}_\sigma$  combined. These  $2^m$  prefix vectors correspond to the corners of a hypercube in the impact time coordinates  $T$ , in which the diagonal  $-\mathbf{1}$  is a trajectory. The corresponding points in the  $W$  domain are no longer a hypercube, but their adjacency relationships are the same, and the cones the span correspond to those in impact times.

Each set of  $m+1$  points  $y_k$  (with  $p_k = \tau^{\tilde{G}}(y_k)$ ) and their images  $y'_k$  under the unit time corner flow (with  $q_k = \tau^{\tilde{G}}(y'_k)$ ) are enough to determine one entire component  $M(\sigma)$ . Fixing  $\sigma \in S_m$ , the map  $v \mapsto \tilde{\phi}(1, v)$  for  $r(v) = \sigma$  is affine and thus uniquely defined by its action on a simplex ([Groff et al., 2003](#)), allowing us to construct a representation of the corner flow. We do so by flowing a simplex of  $v$  values such

---

<sup>4</sup>The inverse  $\tau^{-1}$  is defined for  $m < d$  by restricting it to the  $m$ -plane.

that  $\tau(v) = p_k$ , or equivalently  $\hat{h}(v) = y_k$ , with  $y_k$  which lie on the boundary of the  $\mathcal{K}_\sigma^G$  simplex. The flow computation turns out to be most convenient to do in the  $W$  coordinates, because of a fast computation for integrating the  $\tilde{\phi}^W$  corner flow.

We thus obtain the formula (where, recall, the  $y_i$  and  $y'_k$  are members of  $T_0W$  under  $\mathfrak{I}_W$ )

$$\tilde{M}(\sigma) := \cdot \begin{bmatrix} y'_1 & \cdot & \cdot & \cdot & y'_{m+1} \end{bmatrix} \cdot \begin{bmatrix} y_1 & \cdot & \cdot & \cdot & y_{m+1} \\ 1 & \cdot & \cdot & \cdot & 1 \end{bmatrix}^{-1} \cdot \begin{bmatrix} I_{m \times m} \\ 0 \end{bmatrix} \quad (3.26)$$

The dependence on time is straightforward –  $-1$  in the  $T$  domain. Thus  $p_k$  are in the  $\tau^F(H_i)$  hyperplanes for  $i$  where  $\langle e_i, p_k \rangle = 0$ , and the  $q_k$  they reach after flowing one time unit are in the remaining  $\tau^F(H_{\bar{i}})$  hyperplanes for  $\bar{i}$  where  $\langle e_{\bar{i}}, p_k \rangle = 1$ . If we flow for longer than that, the points merely translate by  $(t-1)F_{\mathbb{1}}$ . If we start flowing from negative times, we similarly translate by  $tF_{-\mathbb{1}}$ . Both these translations do not change any  $D\tilde{\phi}$ , so it is enough to examine  $D\tilde{\phi}(u, w) \mid_{\{0\} \times \text{cl}(\mathcal{K}_\sigma)}$  the restriction to transition sequence  $\sigma$  with transitions starting at time 0. We take the linear part of Eqn. 3.26 ([Groff et al., 2003](#), Eqn. 2), which determines a matrix representation for  $M(r(y))$ .

If  $m < d$ , then we append  $d - m$  points  $\{e_j\}_{j=m+1}^d$  to span the remaining dimensions. We saw in §3.5.5 that  $r_{\tilde{G}} \circ \pi_m = r$ , so if  $m < d$ , for two points  $a = (y_1, \dots, y_m, u_{m+1}, \dots, u_d)$ ,  $b = (y_1, \dots, y_m, w_{m+1}, \dots, w_d)$ ,  $r_{\tilde{G}}(a) = r_{\tilde{G}}(b)$  holds. Thus, the sampled field  $G_b$  is constant in the “ $Q$ ” directions, in the sense that  $\tilde{G}(a) = \tilde{G}(b)$ . In other words, the polytope that is defined by  $\pi_m^{-1}(\mathcal{K}_{r(y)}^G)$  is unbounded in the  $Q$  directions (i.e. is a *prism*), where  $\mathcal{K}_{r(y)}^G$  is the cone in the  $m$ -plane<sup>5</sup> associated to the sequence  $r(y)$  for some point  $y$ .  $\pi_m^{-1}(\mathcal{K}_{r(y)}^G)$  consists of all points which undergo the same transition sequence  $r(y)$ .

Since the transition sequence is the same across the prism, the same linear com-

---

<sup>5</sup>the  $m$ -plane is all  $(a, 0) \in \mathbb{R}^d, a \in \mathbb{R}^m$



ponent  $M(r_{\tilde{G}}) = M(r_{\tilde{G}} \circ \pi_m)$  of the B-derivative (recall, equal to the flow for the sampled vector field) applies to every point in that prism. We could represent all these linear maps by taking  $m!(d-m)$  additional linearly independent points in the last  $d-m$  coordinates in combination with the base simplex (recall that  $G$  is rectified to have the first  $m$  coordinate planes as event surfaces). However, we can reduce this to only  $(d-m)$  additional points by noting that the points  $\{e_j\}_{j=m+1}^d$  are jointly shared between all prisms, as they all project to the origin in the  $m$ -plane (recall 3.5.3). The  $\{e_j\}_{j=m+1}^d$  points are sufficient to span the remaining  $d-m$  coordinates, and are jointly shared between all prisms, so only  $d-m$  additional integrations are required, rather than  $m!(d-m)$ . Intuitively, the last  $d-m$  components are “independent” from the combinatorial complexity of the corner vector field; no discontinuities are ever a result of an offset in those directions.

### 3.5.9 Computing prefix vectors without impact times

To avoid evaluating the impact time functions explicitly while directly producing the necessary pairs  $(y_k, y'_k)$  in  $\hat{h}$ -space we can use the piecewise-affine form of  $\tau^{\tilde{G}}$ .

In the following we employ an extended index notation: for a vector  $V \in \mathbb{R}^n$ , and  $B \subset \{1, \dots, n\}$ , we will use the notation  $(V)_B \in \mathbb{R}^{|B|}$  to refer to the vector consisting of the coordinates appearing in  $B$ , in numerical order. For example,  $(V)_{\{i\}}$  for a singleton set  $\{i\}$  is just the  $i$ -th coordinate of  $V$ , also denoted  $V_i$  or  $(V)_i$ .

Let  $I := \{1, \dots, m\}$ ,  $D := \{m+1, \dots, d\}$ . For each  $A \subseteq I$ , denote by  $\bar{A} := I \setminus A$ , i.e. the set complement of  $A$  in  $I$ . We also take any  $A \subseteq I \cup D$  to denote a corner of the hypercube  $b \in \mathcal{B}^d$  ( $A$  understood from context) whose  $k$  coordinate is 1 if  $k \in A$  and  $-1$  otherwise. This allows us to define  $G_A := \tilde{G}_b(y)$ ,  $y \in (\tau^G)^{-1}(D_b)$ .  $G_A$  is the vector field restricted to the orthant where transitions  $\{N_j : j \in A\}$  have been made, but the transitions  $\{N_j : j \in \bar{A}\}$  have *not* happened.

We define points  $z_A$  by:

$$\begin{aligned}(z_A)_A &:= 0 \\ (z_A)_{\bar{A}} &:= -(F_A)_{\bar{A}} \\ (z_A)_D &:= 0\end{aligned}\tag{3.27}$$

The points  $z_A$  are the points whose  $A$  coordinates and  $D$  coordinates are zero, and whose  $\bar{A}$  coordinates match those of the  $F_A$  vector field. The point  $z_A$  is in  $\cap_{i \in A} N_i$ , and after one unit of time flows into  $\cap_{j \in \bar{A}} N_j$ . This implies that  $\tau(z_A) = \chi_A$ , implying that  $z_A$  is uniquely defined. The uniqueness of  $z_A$  can also be obtained by inspection from the definition above, which is the computational insight that enabled us to develop our algorithm.

Note that the collection of prefix vectors of permutations,  $p_k^{(\sigma)}$  for  $\sigma \in S_m$ ,  $0 \leq k \leq m$  are each associated with a subset  $A \subseteq I$ , because  $p_k^{(\sigma)}$  can be seen as indicator vectors. For  $k = |A|$  the cardinality of  $A$ , view  $(i \in A) \Leftrightarrow (e_i \cdot p_k^{(\sigma)} = 1)$ , or equivalently take  $p_k^{(\sigma)} = \chi_A$ , the indicator vector for membership in  $A$ . Thus the  $2^m$  points  $z_A$  provide the prefix vectors for all  $m!$  permutations  $S_m$ .

We collect the points  $z_A \in \hat{h}(\bar{D}_{-1})$ , which span the  $I$  coordinates of  $W$ . If  $m < d$ , we expand to include the  $D$  coordinates as well, defining a set of triangulation points  $R := \{e_j\}_{j \in D} \cup \{z_A | A \subseteq I\}$ .

The set  $R$  gives rise to the “output set” defined by  $O := \tilde{\phi}(1, R)$  with slight abuse of notation. From the definition of the piecewise affine flow map  $\tilde{\phi}(1, \cdot)$  and of  $z_A$  it follows that  $(\tilde{\phi}(1, z_A))_A = (F_A)_A$  and  $(\tilde{\phi}(1, z_A))_{\bar{A}} = 0$ . From the positivity of all the  $G_b$ ,  $b \in \mathcal{B}$ , it follows that  $(\tilde{\phi}(1, e_j))_I > 0$  elementwise. This allows us to conclude that  $O \subset \hat{h}(\bar{D}_{+1})$ . Our choice of  $t = 1$  is arbitrary; any  $t > 0$  would suffice, we chose  $t = 1$  for convenience.

For a given initial condition  $y(0) \in (-\infty, 0]^d$  subject to the  $G$  induced flow of

the sampled system,  $m + 1$  binary words are always associated to the trajectory: all zeros, for the segment of the trajectory when no surfaces have been crossed, and a binary word arising by changing a 0 to a 1 for each crossed surface (all of which must be crossed by definition for large enough  $t$ ; double crossings resolved arbitrarily), resulting in a lexicographically increasing sequence of words. Taking  $\sigma = s(\tau^{\tilde{G}}(y(0)))$ , this increasing sequence is exactly the prefix vectors  $p_k^{(\sigma)}$ ,  $k = 0 \dots m$ . Ergo, any  $y \in \hat{h}(\bar{D}_{-1})$  gives rise to  $\sigma = s(\tau^{\tilde{G}}(y))$  and through that to  $m + 1$  points  $y_k(y) := (\tau^{\tilde{G}})^{-1}(p_k^{(\sigma)}) \in R$  (in notation we will suppress the  $y$  when clear from context). Furthermore  $y$  is in the cone span of  $\{y_k\}_{k=0}^m$ , which itself consists of all the points which have the same transition sequence as  $y$  under  $\tilde{\phi}^W$ . The piecewise affine map  $\tilde{\phi}^W(1, \cdot)$  takes these  $m + 1$  points into  $O$ , defining an “output” simplex, the points  $y'_k(y) = \tilde{\phi}^W(1, y_k)$ .

Whenever  $m < d$ , we extend  $y_k(y) := e_k$  for all  $y$  and  $k > m$ , ensuring that  $y_k$  span the space  $W$ . Correspondingly, for  $k > m$ ,  $y'_k(y) := \tilde{\phi}^W(1, e_k)$ .

The unique affine map (eqn. 3.26) defined by mapping the initial cone of  $y_0, \dots, y_d$  to the output cone of  $y'_0, \dots, y'_d$  is a triangulation . of the flow  $\tilde{\phi}^W$  restricted to the cone  $\mathcal{K}_\sigma^G$  that has  $\sigma$  as its transition order. This is the **key insight** that enables our algorithm to produce a dramatic speedup — finding a small number of points whose images under the flow triangulate the B-derivative of that flow.

By taking the linear part of this map, we obtain (recall, domain to itself)  $D_y \tilde{\phi}^W(t, \cdot)$  <sup>6</sup> of the corner limit flow at  $\hat{h}(\rho) = 0$ , taken for  $y(0)$  small enough such that  $\tilde{\phi}^W(t, y(0)) > 0$  element-wise. Pulling back through the coordinate change  $\hat{h}$ , we can obtain the desired approximation in the original coordinates of the domain  $U$ . It is also the B-derivative of  $\phi$ , because  $\tilde{\phi}$  and  $\phi$  have the same B-derivative.

Summarizing the above discussion, we have theorem III.10.

**Theorem III.10.**  $\forall \sigma \in S_m$ , let  $\mathcal{K}_\sigma \subset T_0V$  be the cone partition of  $T_0V$  defined in

---

<sup>6</sup>Recall, we fixed  $t = 1$



cone decomposition s.t.

$$J_\sigma = \text{cone}(\{z_{\eta(k,\sigma)}\}_{k=0}^m) \oplus \text{span}\left(\{e_j\}_{j=m+1}^d\right)$$

8

4. (§3.5.9 the impact time of  $z_A$  is  $\chi_A$ )  $\forall k \in I, \sigma \in S_m, \tau^G(z_{\eta(k,\sigma)}) = \chi_A = p_k^{(\sigma)}$
5. (§3.5.9 factorial only in  $m$  coordinates)  $\forall y \in J_\sigma, r_{\tilde{G}}(y) = r_{\tilde{G}} \circ \pi_m(y)$ , so that,  
 $D\tilde{\phi}|_{J_\sigma} = \tilde{M}(r_{\tilde{G}})$ , where  $\tilde{M}(r_{\tilde{G}})$  is the linear part of the matrix computed with  
the formula  $\tilde{M} = A^{-1}B$ , where

$$A := \begin{bmatrix} \tilde{\phi}(1, z_{\eta(1,\sigma)}) & \cdots & \tilde{\phi}(1, z_{\eta(m,\sigma)}) & \tilde{\phi}(1, e_{m+1}) & \cdots & \tilde{\phi}(1, e_d) \end{bmatrix}$$

$$B := \begin{bmatrix} z_{\eta(1,\sigma)} & \cdots & z_{\eta(m,\sigma)} & e_{m+1} & \cdots & e_d \\ 1 & \cdots & 1 & 1 \dots & 1 & \end{bmatrix}$$

Thm. III.10 is not ordered as the sections were. Instead, it is ordered in the logical dependence of the computation process. First, we identify that cones are adequate to describe the linear components of  $D\tilde{\phi}$ . Then, we represent those cones' span with an explicit, useful, basis in  $y$ -coordinates for  $\{J_\sigma\}_{\sigma \in S_m}$ . After we compute all linear pieces of  $D\tilde{\phi}$ , we pull it back via  $h$  to original  $x$ -coordinates. We then conclude with  $D\phi|_\rho = D\tilde{\phi}|_0$ .

---

<sup>8</sup> i.e.,

$$J_\sigma = \text{int} \left( \left\{ \sum_{k=0}^m \beta_k z_{\eta(k,\sigma)} \mid \beta_k \geq 0 \right\} + \left\{ \sum_{j=m+1}^d \beta_j e_j \mid \beta_j \in \mathbb{R} \right\} \right),$$

### 3.6 Computational Procedures

In essence, our algorithm consisted of lazy evaluation of  $x \mapsto y_k(\hat{h}(x))$  and  $x \mapsto y'_k(\hat{h}(x))$  for all values of  $x$  in  $\bar{D}_{-\mathbb{I}}$  and  $0 \leq k \leq m$ . Given these, it is simple to compute the  $m!$  linear maps that define  $D\phi^W(0, 0; u, v)$  and pull this back to  $\mathbf{T}_\rho U$  using  $D\hat{h}$ .

Our input was a full description of the event-selected system: (1) A function  $h : U \subset \mathbb{R}^d \rightarrow W \subset \mathbb{R}^m$  defining the transition manifolds; (2) A vector field  $F : \mathcal{B}^m \times U \rightarrow \mathbf{T}U$ . We assumed w.l.o.g that  $\rho = 0$ .

From  $h$  we computed the matrix  $Dh : \mathbf{T}_\rho U \rightarrow \mathbf{T}_0 W$ , and  $F_b = F(b, \rho) \in \mathbf{T}_\rho U$ . This is the corner vector field, because  $F(b, \cdot)$  is continuous by assumption. Next we produced the Jacobian  $D\hat{h}$  by extending  $Dh$  orthogonally to its row-space. We did this by using a full SVD of  $Dh$ , and replacing the zero singular values with ones to extend  $h$  to  $\hat{h}$  with a linear map on what is the kernel of  $Dh$ . Using  $D\hat{h}$ , we computed  $G_b \in \mathbf{T}_0 W$ , i.e. the function  $G_b := D\hat{h} \cdot F_b$ .

#### 3.6.1 Fast integration of the corner flow

The relationship  $\tilde{\Xi} = \Xi$  allows us to build and exploit one of the key contributions of this paper: a fast integrator for the corner flow  $\tilde{\phi}$ , based on  $\phi(t, x) \approx D\hat{h}^{-1} \cdot \tilde{\phi}^W(t, \hat{h}(x))$ . The fast integrator took as an input  $y \in W$  and  $T \in \mathbb{R}$ , and produced  $(t^i, b^i, y^i) \in [0, T] \times \mathcal{B}^m \times W$  the sequence of impact times, W orthants, and impact points that the initial condition  $(t^0, b^0, y^0)$ ,  $t^0 = 0$ ,  $y^0 = y$  encounters on its way to its final state  $(t^k, b^k, y^k)$ ,  $t^k = T, y^k = \tilde{\phi}^W(T, y)$ . Each  $(t^i, b^i, y^i)$  except for the first and last are a transition manifold crossing that satisfied  $b^i = b^{i-1} + e_j$  for  $j$  such that  $y^i \in N_j$  is the index of the manifold being hit at time  $t^i$ .

To perform this fast integration, we began by initializing  $t^0 = 0$ ,  $y^0 = y$  and  $b^0 = \chi_{y \geq 0}$  – the indicator of which coordinates of  $y$  are non-negative. To compute  $(t^{k+1}, b^{k+1}, y^{k+1})$  from  $(t^k, b^k, y^k)$  Next examined  $y^k/G_{b^k}$  element-wise, and let  $0 \leq$

$j \leq m$  be the index of the smallest non-negative element among these ratios (breaking ties arbitrarily). The ratio  $(y^k/G_{b^k})_j$  is the time to the next impact under the flow. If it was less than  $T - t^k$ , then we took

$$\begin{aligned} y^{k+1} &:= (y^k/G_{b^k})_j G_b + y^k \\ t^{k+1} &:= (y^k/G_{b^k})_j + t^k \\ b^{k+1} &:= b^k + e_j. \end{aligned} \tag{3.28}$$

If it was more than  $T - t^k$ , then we took

$$\begin{aligned} y^{k+1} &:= (T - t^k)G_b + y^k \\ t^{k+1} &:= T \\ b^{k+1} &:= b^k + e_j \end{aligned} \tag{3.29}$$

and the integration was ended. In essence, the variables  $b^k$  is maintaining a binary count that indicates if a transition has occurred or not, resolving the need to detect event crossings numerically.

### 3.6.2 Computing the B-derivative of the flow

To compute B-derivatives efficiently, we used dynamic programming to cache pairs  $(r, o) \in R \times O$  such that  $\tilde{\phi}(1, r) = o$ , in a mapping indexed by the set  $\mathcal{B}^m \cup \{m + 1, \dots, d\}$ . Given an initial point  $x \in \bar{D}_{-1}$ , we transformed it into  $y = h(x) \in W$ , and integrated using the fast corner flow integrator to obtain  $\{(t^i, b^i, y^i)\}_{i=0}^m$ . The corners  $b^i$  were used as indices for the  $z_A \in R$  and thus for lookup in this cache. If a corner  $b$  was not found in the cache, we calculated the  $z_A$  for  $b = \chi_A$ , then rapidly integrated  $h(z_A)$  and used the times  $t^i$  and the  $b_i$  to compute  $x(1) = x(0) + \sum_{k=0}^{m-1} (t^{k+1} - t^k) F_{b_k}$ . This gave us the result needed for caching:  $b \mapsto (r, o) = (x(0), x(1))$ .

Given the  $m + 1$  point pairs  $\{(r_k, q_k)\}_{k=0}^m$ , and the  $d - m$  point pairs obtained from integrating the initial conditions  $e_j \in W$  with  $j > m$ , and cached with the mapping keys  $m + 1, \dots, d$  the unique affine map taking these  $r_k$  to  $q_k$  is the piece of B-derivative that applies to the initial condition  $x(0)$ . It can be computed using ordinary least-squares.

### 3.6.3 A note on computational complexity

The fast corner flow integration consists of finding a minimal element among  $m$  elements,  $m$  times, and then performing other element-wise operations a uniformly bounded number of times – giving a total running time which is  $\mathcal{O}(m^2)$  on a sequential machine and  $\mathcal{O}(m)$  on a parallel one. This is followed by a least-squares calculation on  $d + 1$  vectors of dimension  $d + 1$ , giving a  $\mathcal{O}(d^3)$  which dominates the running time. The cache size is  $\mathcal{O}(2^m)$ .

However, if speed is of the essence, a cache of size  $\mathcal{O}(m!)$  can be used together with the fast integrator, allowing each of the least-squares problems to be computed at most once. After the cache is complete, each new point then requires only one pass of the fast corner flow integrator — but at the expense of significantly greater storage.

By comparison, a conventional central differences numerical Jacobian of the flow map would require a minimum of  $d + 1$  *integrations* per each of the  $m!$  transition sequences, followed by an  $\mathcal{O}(d^3)$  least-squares calculation. The number of steps per integration would be at least equal to  $m$ , and each step would be  $\mathcal{O}(d)$  on a sequential machine;  $\mathcal{O}(1)$  on a parallel one. In practice, however, general purpose hybrid system integrators are likely to take far more than the minimal number of steps. Even assuming each set of  $d + 1$  integrations is computed in parallel, this would require  $\mathcal{O}(d^3 m!)$  whereas our algorithm requires only  $\mathcal{O}(d^3 2^m)$ .



### 3.6.4 Validation

We implemented the above algorithm in `Python-2.7.5` augmented with the `Numpy` and `Scipy` numerical processing toolkits.

### 3.6.5 Piecewise Constant Systems

The first system we measured the performance of our implementation upon was a 2-dimensional piecewise constant vector field.

$$F_{\text{sign}(x)} = \nu \mathbf{1} - \delta \text{sign}(x)$$

It is clear that this system is event selected for  $\delta < \nu$ , and that the coordinate axes are event surfaces. The number of event surfaces coincides with the dimension, that is,  $m = d$ . In §8.1.1 of [Burden \*et al.\* \(2016\)](#), it is shown that  $\forall x \in D_{-\mathbf{1}}$ ,

$$D_x \phi(0, 0_2) = \frac{\nu - \delta}{\nu + \delta} \mathbf{I}_2$$

For choices  $\nu = 1, \delta = 0.6$ , this is the diagonal matrix

$$D_x \phi(0, 0_2) = \begin{bmatrix} 0.25 & 0 \\ 0 & 0.25 \end{bmatrix}$$

Our algorithm produces the same result to numerical precision.

$$L(x) = \begin{bmatrix} 0.25 & 0 \\ -3.4 \cdot 10^{-17} & 0.25 \end{bmatrix}$$

The largest singular value of  $D_x \phi(0, 0_2) - L(x)$  is  $4.4 \cdot 10^{-7}$ .

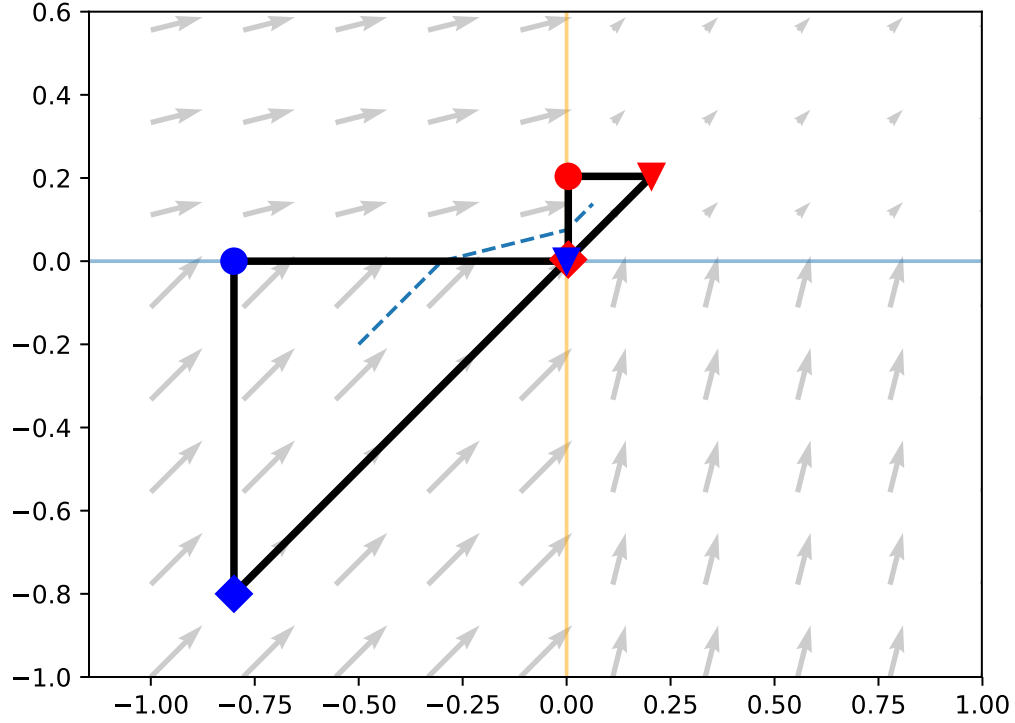


Figure 3.3: A quiver plot of the  $F$  in the plane for  $\nu = 1$ ,  $\delta = 0.6$ . In this case the original and corner flows coincide. The dashed line is the segment of a solution - it's transition sequence determines the value of  $D\tilde{\phi}(0, 0; u, v)$  on the drawn simplices. The lower simplex is mapped to the upper one. Vertices with markers of shared type are paired by  $\tilde{\phi}(1, \cdot)$ , and thus, by  $D\tilde{\phi}$ .

### 3.6.6 Stable Second Order Oscillator

The second application we considered to verify accuracy was another event-selected system with a known, closed form B-derivative. Let  $(x, \dot{x}) \in \mathbb{R}^{2d}$ , with equations of motion

$$F(x, \dot{x}) = \begin{bmatrix} \dot{x} \\ \alpha \mathbf{1} - \beta \dot{x} - \delta \text{sign}(x) \end{bmatrix}$$

$F$  has event surfaces of the first  $d$  standard coordinate planes in  $\mathbb{R}^d$ . It can be shown ([Burden et al., 2016](#), Eqn. 104) that the B-derivative of the flow at  $(t, x, \dot{x}) = (0, 0, \nu \mathbf{1})$

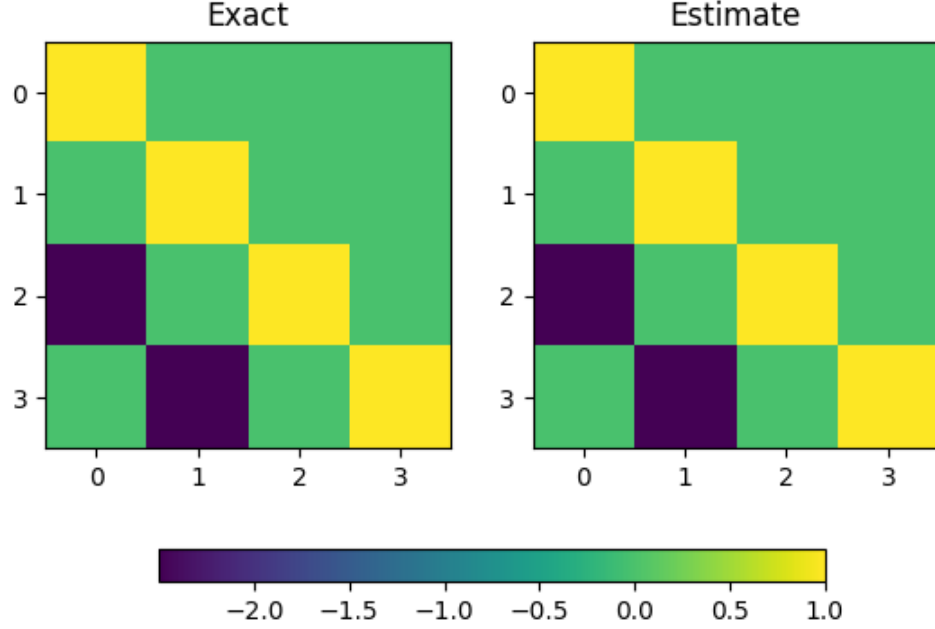


Figure 3.4: Heat maps comparing the results from computing the B-derivative with our algorithm, versus numerically evaluating the known exact result.

is :

$$D_\phi(t, x) = \begin{bmatrix} \mathbf{I}_d - \frac{2\delta}{\nu\beta} (1 - e^{-\beta t}) & \frac{1}{\beta} (1 - e^{-\beta t}) \\ -\frac{2\delta}{\nu} e^{-\beta t} \mathbf{I}_d & e^{-\beta t} \mathbf{I}_d \end{bmatrix}$$

We numerically evaluated the above statement at  $t = 0$ ,  $\delta = 0.5$ ,  $\alpha = 1$ ,  $\beta = .4$ , visualized as the heat map in Fig. 3.4.

### 3.6.7 Planar Rigid Body with Contacts

The final example we present is a single almost-planar rigid body with two massless legs colliding with the ground. We wish to compute the B-derivative around the point  $\rho$  of simultaneous contact of both legs with the ground. We elect the configuration space  $\mathfrak{se}(3)$  evolving under the Newton-Euler equations ([Murray, 2017](#)), shown in

Eqn. 3.30) with gravity

$$\begin{bmatrix} mI & 0 \\ 0 & \mathcal{I} \end{bmatrix} \begin{bmatrix} \dot{v}^b \\ \dot{\omega}^b \end{bmatrix} + \begin{bmatrix} \dot{\omega}^b \times m v^b \\ \dot{\omega}^b \times \mathcal{I} \omega^b \end{bmatrix} = F^b \quad (3.30)$$

$v^b$  is the translational body velocity in the initial frame, while  $\omega^b$  is the rotational in the same frame.  $m$  is the mass of the rigid body,  $\mathcal{I}$  is the inertia tensor.  $I$  is the identity matrix, and  $F^b$  is a wrench applied at the center of mass. We use the term “almost” above, as despite being numerically simulated as a 3D rigid body, we confine our rigid body to the  $z = 0, \dot{z} = 0, \omega_x = 0, \omega_z = 0$  plane using a strong restoring force to stabilize the plane.  $(x, y)$  identify the center of mass location, and  $\omega = \omega_y$  is the current angle, reducing our system to being on  $\mathfrak{se}(2)$ , only having the  $x, y, \omega_y$  degrees-of-freedom.

Each leg  $l_i$  is a fixed vector when written body coordinates. From the current  $l_i(t)$  in contact with the ground (which is implicitly a function of state), a force is generated that acts on the rigid body. Each leg is massless, has no inertia, operating as a pure force generator with no internal states. The arrangement is intended to represent a chair or stool with rigid legs impacting a slightly elastic ground surface. The “seat” of the chair is represented by a collection equal-mass points in a rigid configuration, i.e. the inertia tensor  $\mathcal{I}$  is diagonal.

For simplicity of implementation, and to insist that the system is event selected, we impose the constraint that once a leg has made contact with the ground, it is irrevocably attached with a constant attachment point (no sliding or liftoff.) To account for this constraint, the force law for each leg is given as follows.

When a leg makes contact with the  $y = 0$  plane, a distinguished point  $l_i^c$  is cached, defining a vector between the current foot position  $l_i(t)$ , which is still allowed to vary. The force generated via this contact (it is identically zero prior to contact) is  $F(l_i(t)) := k_p(l_i(t) - l_i^c) - k_d(\dot{l}_i(t))$ . This intuitively is a spring-damper pulling

the current foot position back to the initial point of contact. We define functions  $h^i(x(t)) = -l^i(t)_y$  as event functions, that is, the current foot height is defined as an event function.

We recall that being event-selected is a *local* property, and we are attempting to compute the B-derivative at the point of simultaneous contact. If the legs are sufficiently stiff, the chair will rebound on impact, violating the assumption that  $\forall i, t, L_f h^i(t) > 0$ , but for some choices of the inertia matrix  $I$ , and the spring constants  $k_p$ , and  $k_d$ , there are a range of initial conditions which the the system is event selected – the foot height evolves monotonically for a time and the second foot will impact the ground prior to the first foot reaching its nadir.

Rather than use closed-form expressions, we elect to persue this example completely numerically; as a result of lacking closed-form expressions for  $D\phi(0, \rho; 0, v)$ , we instead will determine the algorithm’s accuracy by comparing the resulting matrices to the numerical Jacobian of the flow.

Recall from above that the  $D\phi(0, \rho; 0, v) = M(r(x)) \cdot v$ . so that for the non-linear flow  $\phi(t, x : u, v)$ ,

$$\lim_{(t,x) \rightarrow (0,\rho)} D_x \phi|_{(t,x)} \cdot v = M(r(x)) \cdot v,$$

and  $\phi(t, x)$  is smooth for times large enough such that  $x(t) \in D_{+1}$ . Ergo, the numerical Jacobian at a point  $x(0) \in K_{r(x)}$  is arbitrarily close to  $M(r(x))$  as  $\|x(0)\|$  becomes arbitrarily small. While the components of each matrix are continuous, and this can be used to more precisely define “close” in the preceding statement, we elect to use a norm that compares matrices as operators directly; precisely, by  $\|M \cdot D\phi^{-1} - I\|_2$  for the operator 2-norm, which equivalently gives the largest singular value of  $(M \cdot D\phi^{-1} - I)$ . This value will be zero if the two matrices are identical. Since the limit of the operators converges, the singular values must as well, so we expect to be make to make this norm small.

We have taken  $m = i = 2$  for this example, and since  $SE(3)$  is six-dimensional,

so we are in the  $m < d$  regime where number of transition manifolds is smaller than the dimension of state space. We numerically solve for a matrix  $Q$  such that  $\hat{h} := (-l_y^1, l_y^2, Q)$  is a local diffeomorphism around the point  $\rho$ . In principle, we argued above that this choice does not effect the resulting computation. In practice we chose  $Q$  on a similar scale as  $\nabla h$  to avoid numerical issues. Displayed in Fig. 3.5 is a comparison of the numerical Jacobian and B-derivative representation for the initial condition  $(-0., 2.57, -0.001., -0., -8.2, -0.)$  This condition corresponds to the center-of-mass starting at an initial height of 2.57 units, with the rigid body slightly tilted. The “right leg” touches down, then the “left leg” touches down after. The largest singular value of the difference operator is 0.093. When we evaluated conventional Jacobians taken at different points and scored them pairwise via the same metric, we found 0.093 to within the distribution of values, suggesting the the non-linearity of the system is dominating the lower bound of the error.

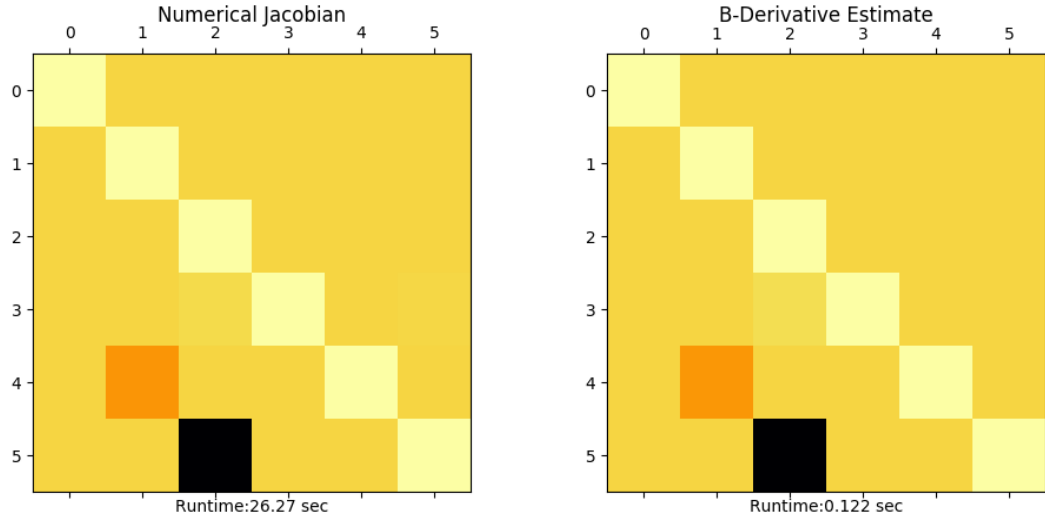
The ratio of the run-times between the conventional finite-difference method and the new algorithm on our workstation is 214; the algorithm presented here is over  $\times 200$  faster than using central differences. It is worth noting that the B-derivative algorithm computes *every* prefix point - the 214x factor is only for the Jacobian at a *single point* — including all four cones (at least 2 Jacobians) <sup>9</sup> indicates that the new algorithm is  $\times 400$  faster versus central differences when computing the total derivative.

### 3.7 Summary of Numerical Methods

Multi-contact simulations in robotics presumably face a factorial growth in their complexity due to all possible contact sequences appearing in their hybrid dynamics. For event-selected hybrid systems, which govern many multi-limb and multi-finger

---

<sup>9</sup>Since the ‘no contact’ initial conditions are uninteresting, and the ‘simultaneous touchdown’ is measure zero, we exclude them for this comparison.



(a)

(b)

Figure 3.5: A comparison of the numerical Jacobian and B-derivative representation for the initial condition  $(-0. , 2.57, -0.001. , -0. , -8.2 , -0.)$ .

contact problems, the local properties of the flow can be determined by a first order approximation called the Bouligand derivative (or “B-derivative”).

Here we presented an algorithm that has a combinatorial speed improvement relative to naive numerical approaches thanks to its ability to employ a minimal collection of triangulation points which represents all linear pieces of the B-derivative. Furthermore, our algorithm includes and employs a high-speed “corner flow integrator” which can integrate a piecewise constant vector field extremely quickly using a finite number of steps. The improvement in run-time allows the numerical integration of the variational equation of event-selected systems to be conducted much more rapidly, especially when the  $m \ll d$ .

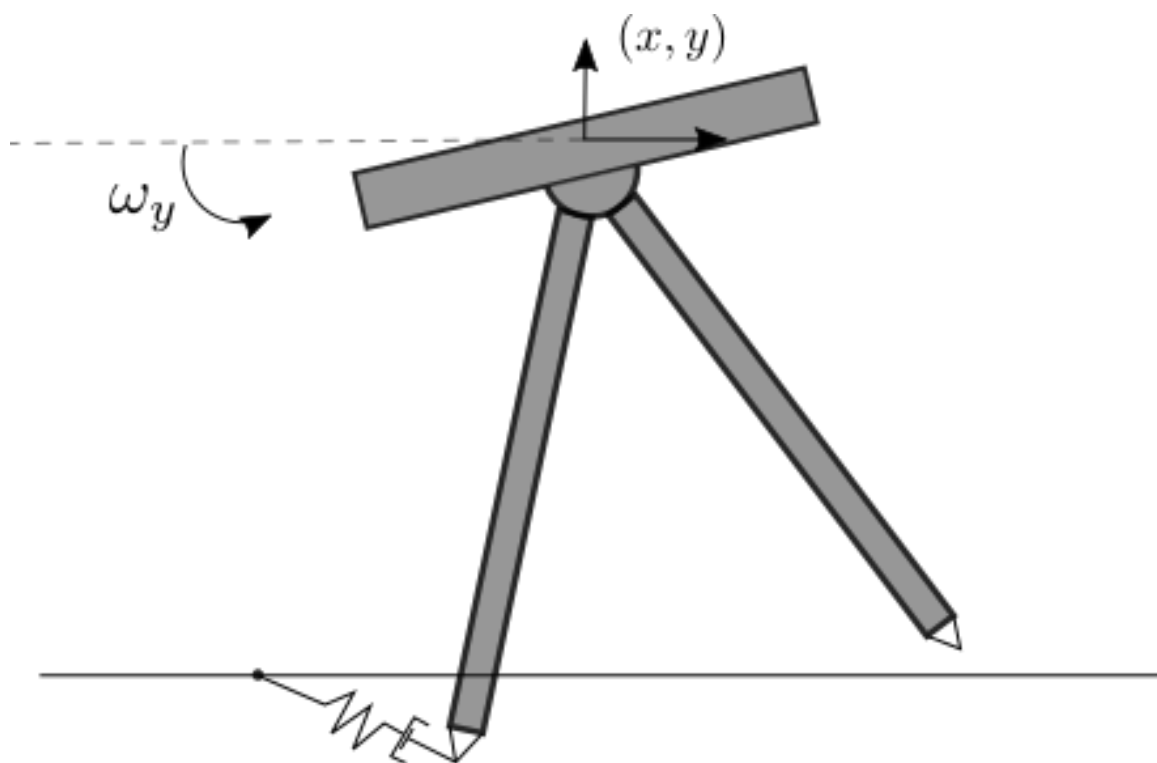


Figure 3.6: The chair model we simulated. The spring-damper is a restoring force that was generated based on the contact point of the chair. In the air, the spring-damper is undefined. Once a leg touched down, the spring-damper instantaneously appeared.



### 3.8 Piecewise-Constant Feedback

We now shift gears, and consider how to employ  $EC^r$  vector fields to the stabilization of multi-legged gaits. An example (which we will abridge here) from ([Burden et al., 2016](#), 8.1) we will exploit considerably in the sequel is the synchronization of first-order phase oscillators by piece-wise constant feedback. The resulting autonomous system is an  $EC^r$  vector field whose stability properties can be evaluated via Cor. III.8.

Consider the following collection of  $n$  identical oscillators with control inputs  $u_i$ , for  $i = 1, \dots, n$ ,

$$\dot{q}_i = \omega + u_i(q) \quad (3.31)$$

where  $q_i \in S^1$ , so that the state space is an  $n$ -dimensional torus. We want a piecewise constant feedback that locally exponentially stabilizes the orbit,

$$\Gamma = \{q \in T^n : \forall i, j \in \{1 \dots, n\} : q_i = q_j\}$$

That is, Let  $U_\delta \subset Q = \mathbf{T}^n$ , for  $\delta > 0$ , be the following set, where  $\pi : \mathbb{R}^n \rightarrow Q$  is the canonical quotient map, considered as a covering map of  $\mathbf{T}^n$ , which is the standard unit cube in  $\mathbb{R}^n$  under the equivalence relationship identifying opposing faces with matching orientation.

$$U_\delta := \left\{ q \in Q : \exists x \in \pi^{-1}(q) : \|x\|_1 \leq \frac{\delta}{n} \right\}$$

$\delta$  must be chosen sufficiently small so that  $U_\delta$  is evenly covered, i.e. -  $\pi_{\pi^{-1}U_\delta}$  is a homeomorphism onto its image.

Define  $\text{sign} : \mathbb{R}^n \rightarrow \{-1, 1\}^n$  by :

$$\forall x \in \mathbb{R}^n : \pi_i \circ \text{sign}(x) = \begin{cases} -1 & x_i < 0 \\ 1 & x_i \geq 0 \end{cases}$$

where  $\pi_i$  is the projection on to the  $i$ -th coordinate. The feedback term we use is given by:

$$\forall q \in Q : u(q) := \begin{cases} -\omega_{SL} \text{sign} \circ \pi^{-1}(q), & x \in U_\delta; \\ 0, & x \in Q \setminus U_\delta; \end{cases}$$

where  $\omega_{SL} \in (0, \omega)$ . Recall that  $w$  is the drift velocity of eqn. (3.31).

It can then be shown ([Burden et al., 2016](#), eq. 96) that the Poincaré map has Bouligand derivative

$$DP(\rho) = \frac{\omega - \omega_{SL}}{\omega + \omega_{SL}} I_{n-1}$$

By Cor. III.8, it is a contraction, and thus  $\Gamma$  is exponentially stable.

The above situation is depicted graphically in Fig. 3.7, for the case of  $n = 2$ . Imagine an example execution for  $q_1(0) < 0, q_2(0) < 0$ , nearby the  $q_1 = q_2$  set. Each state  $q_i$  proceeds at a nominal velocity  $\omega$  until the state component with the lowest positive impact time crosses its coordinate hyperplane in the standard arrangement, where it then reduces speed to  $\omega_{SL} < \omega$ . By doing so, the complementary state progresses in phase against it. The relative velocity between states changes from 0 to  $\omega - \omega_{SL} > 0$ , reducing the difference between  $q_1$  and  $q_2$ . Once both  $q_1$  and  $q_2$  transition, and both velocities are  $\omega_{SL}$ , their relative velocity returns to 0, producing a finite amount of contraction. When the solution exits  $U_\delta$  through the upper quadrant in positive time, the vector field is set back to  $(\omega, \omega)$  via the equivalence relation (note that this means the vector field is also discontinuous there), preserving the difference between  $q_1$  and  $q_2$ . The contraction process repeats in forward time infinitely often, leading to  $\|q_1 - q_2\| \rightarrow 0$  as  $t \rightarrow \infty$ .

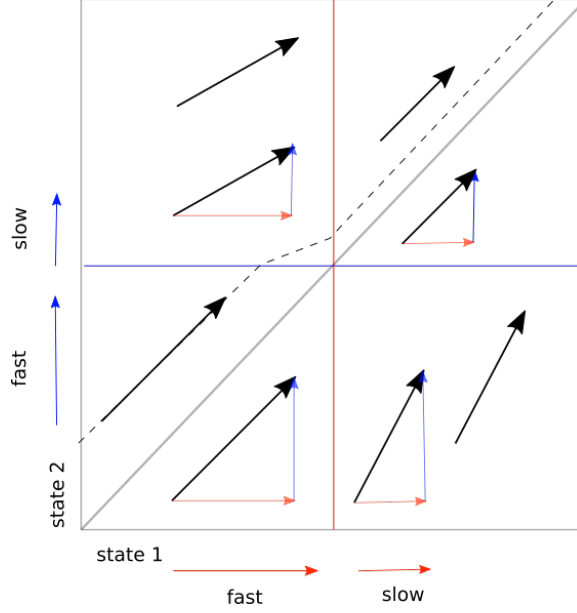


Figure 3.7: An illustration of locally stabilizing the diagonal.

For purposes of the sequel, we refer to this configuration (Fig. 3.7) of piecewise constant vector fields stabilizing the diagonal with event surfaces of the standard arrangement and an exit section as a “device”.

### 3.8.1 Consequences for Implementation

We make a second important remark relevant to robotics at this juncture. Changes in the vector field occur *only* at event surfaces – in between them, the vector field is constant. Moreover, evaluating if  $i$ -th coordinate axis has been crossed is determined only via the one coordinate  $q_i > 0$ . In the context of actuating servomotors, we take advantage of this kind of structure to simplify the amount of information that is required to implement a desired  $EC^r$  vector field. Since a standard system structure for hardware is to have a centralized agent connected to a network of servo modules, the above example suggests that the only state information that needs to be reported from a servo module with a first-order hold (a first-order hold is exactly tracking a piecewise-constant vector field) to the central agent is when an event surface is crossed. Additionally, some control decisions do not depend on the total state - servo

module with state coordinate  $q_i$  can autonomously take the control action associated to crossing the hyperplane  $q_i = 0$  without needing know the global state  $q$ .

For example, it may be again instructive to consider the time-dependence of a trajectory of the “device” in Fig. 3.7. Assume that the initial condition  $x_0$  lay in the  $(-1, -1)$  quadrant, below the diagonal. Since we are assuming the vector field is  $EC^r$ , the trajectory will intersect the  $y$ -axis, and cross into the lower right quadrant. By the feedback given in Eqn. (3.8), the vector field *only changes its horizontal component*. Ergo, if each servomotor independently reduced its velocity to  $\omega_{SL}$  based on knowledge of only its own state, it would still be a correct implementation of Eqn. (3.8). Only the exit condition given by  $\partial U_\delta$  depends simultaneously on the value of  $q_1$  and  $q_2$ .

In other words, the event surfaces partition the state space into regions. In order to determine what control action should be taken, the controller needs to know what region of state space the current state occupies. The temporal ordering of a solution curve mandated by being  $EC^r$  reduces the need to use the full state to determine what vector field applied at a given state should be. We see that we could, for example, use the *history* of a given execution to implicitly determine the vector field except at the exit surface. We see no obvious way to extend this kind of distribution of control to every event surface – eventually global knowledge is needed for contraction. We may make an improvement on the information requirements of even the exit surfaces, however. If we further more restrict the class of exit surfaces to be products as well, where we restrict *every* event surfaces to be a collection of hyperplanes, the entire state space can be partitioned into boxes. In this case, only a flag indication crossing would be reported, rather than a numeric value.

While we admit this may restrict the class of attractors for the time being, (we will address stabilizing generic polygonal curves using this strategy in the sequel), it is clear that the contraction will still occur for trajectories sufficiently local to the

diagonal. Ergo, to detect the exit surface, each motor can report crossing it to a central agent. The agent can then perform a logic AND on each report – if the AND returns true, it can update the first-order holds of the servomodules.

We have described that implementing a piecewise constant vector field on the  $n$ -torus in hardware can be accomplished with a distributed controller with a finite number of events that require global information exchange. Such a system architecture is radically distinct from classical servoing, where a central agent gets frequent/periodic updates from each servo at some known sampling frequency, and produces control actions that are simultaneously transmitted to the servomodules.

Beyond the hardware implications, the event occurrences are *essential* to the genesis of a stable attractor. Any individual vector field trivially has no attractors as a constant vector fields have affine solutions with no lower dimension  $\omega$ -limit. In light of 3.6, the saltation matrices may be the only contractive elements in the B-derivative of the flow – the smooth fundamental matrices have a unity operator norm. We tie this more concretely to robotics in the sequel, but contraction arising from event functions invokes the idea of using the ground impacts themselves of a robot to produce a gait. Without the “in-situ” impacts, no lower dimensional behavior would be observed.

### 3.8.2 General Polygonal Curves

The preceding example of using piecewise constant feedback to stabilize the diagonal of the flat torus employs a structure that is generic to any polygonal curve, which we define to be A sequence of knot points  $V := \{p_1, \dots, p_n : i = 0, \dots, n\}$  connected by the line segments  $\{tp_{i+1} + (1 - t)p_i : t \in (t_j, t_{(j+1)}), j = 0, \dots, n - 1\}$ . If we were interested in rendering the locus of the curve attractive, we could proceed in the following fashion. Given the point  $p_i$ , if we choose a sufficiently small neighborhood  $U_i$  such that  $p_j \notin U_i$  for  $j \neq i$ , we place a “device” at the point  $p_i$ . If we did this for all  $i$ ,

we would generate a vector field that produces a finite amount of attraction towards the desired locus each time a trajectory passed nearby a knot point. If the polygonal curve was a *closed* polygonal curve, as time went to  $\infty$ , any initial condition starting locally off the target locus would exponentially collapse to it by Cor. III.8.

The above is only the essential idea of generating a stabilizing controller. It clearly does not determine a vector field, even locally around the target locus, as either the vector field is undefined outside of the  $U_i$ 's, or the vector field is multi-valued if  $U_j \cap U_i \neq \emptyset$ . We refine it into the following algorithm, which is restricted to a local tube around monotonic curves. Let  $V := \{p_1, \dots, p_n : i = 0, \dots, n\}$ ,  $|V| \geq 3$ , define a monotonic polygonal locus in  $\mathbb{R}^d$ .  $\forall i, j, p_{i+1}^j > p_i^j$ , where  $p_i^j := \pi_j \circ p_i$  (the  $i$ -th component). We will produce a local rectifying change of coordinates  $C_{p_i} : D_{p_i} \rightarrow \mathbb{R}^d$ , for  $D_{p_i}$  to be given, defined around each  $p_i$  that transforms the locus to the diagonal with event surfaces at the standard arrangement. Given a point  $x$ , we deduce the ordering  $I \subset \{1, \dots, n\}$  such that for each coordinate  $x^j$ ,  $p_{i-1}^j < x^j < p_i^j$ . If  $\exists j : (i_j - i_{j-1}) > 1$ , we reject the point  $x$  as being outside the domain of  $C_{p_i}$ . All such  $x$  that satisfy this locality condition is the domain  $D_{p_i}$  of  $C_{p_i}$ , and will be a union of rectangular solids where  $\forall x \in D_{p_i}, x^j > p_{i-1}^j$ , so that  $C_{p_i} : D_{p_i} \rightarrow \mathbb{R}^d$  as, for each coordinate  $j$ ,  $\pi_j \circ C_{p_i}(x) = (x^j - p_i^j) / (p_i^j - p_{i-1}^j)$ . This piecewise smooth homeomorphism rectifies the rectangular partition around the point  $p_i$  to the canonical arrangement. We then stipulate the vector field to be the canonical device in the rectified coordinates, and map it back to  $D_{p_i}$  by  $C_{p_i}^{-1}$ . Shown in Fig. 3.8 and Fig. 3.9 are example simulation results for arbitrary monotonic curves.

Each node  $p_i$  produces only finite contraction towards the polygonal curve. For a desired locus to be asymptotically stable, an infinite number of transitions are required. While this is clearly the case for periodic systems (as shown above in III.8), some other method would have to be employed for non-periodic curves.

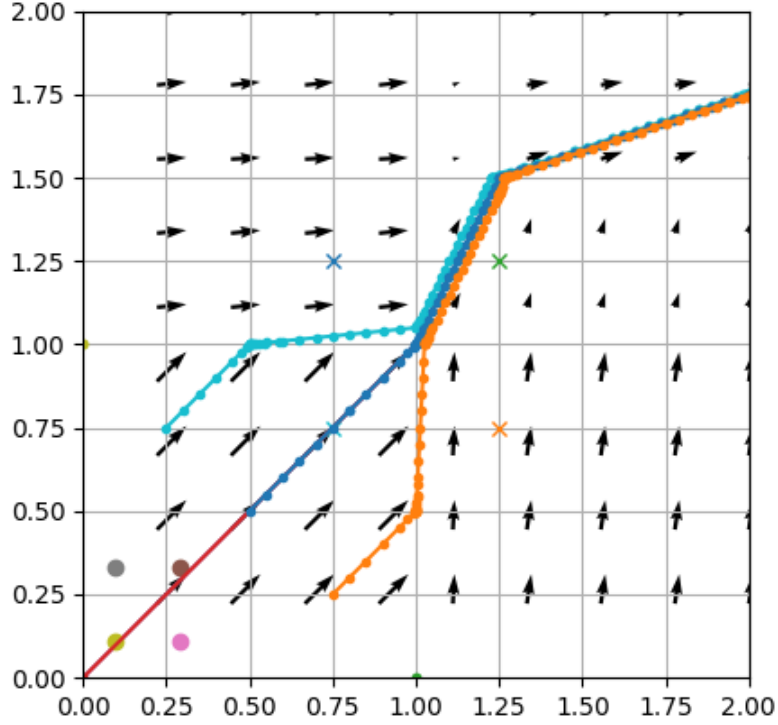


Figure 3.8: Monotonic polygonal curve in the plane stabilized with canonical devices through a rectifying change of coordinates.

### 3.9 ESS and RHex

The RHex robot is a well-known dynamically mobile hexapedal robot ([Galloway et al., 2010](#); [Saranli et al., 2001](#)), with six non-jointed compliant limbs attached to a central body through a single direct-drive rotatory joint, shown in Fig. 3.10

In collaboration with Kod\*Lab of University of Pennsylvania, we constructed a RHex-type robot to illustrate that event-selected piecewise constant feedback is a sufficient mechanism to generate gaits, and have an implementation that is parsimonious in communication complexity compared to conventional reference-tracking controllers. The theoretical results of  $EC^r$  systems will be employed in demonstrating resiliency against uncertain ground contacts and realistic implementation constraints.

A persistent forward motion with the RHex robot is most obviously achieved by

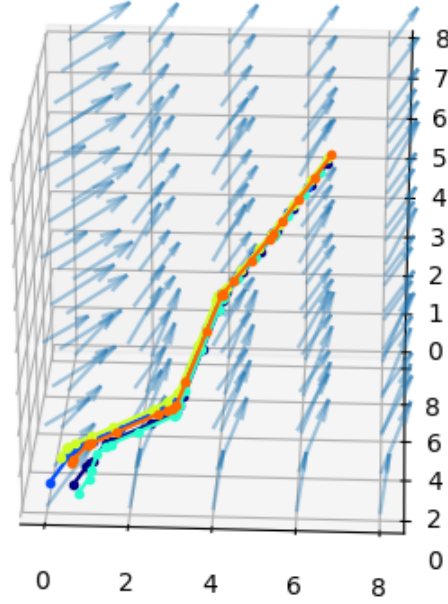


Figure 3.9: Monotonic polygonal curve in 3-space stabilized with canonical devices through a rectifying change of coordinates.

requiring a governing control scheme to drive the limbs periodically – a gait such as a walk or trot. A well-established ([Saranli et al. \(2001\)](#) and references therein) gait in the RHex community is the alternating tripod gait, where groups of three limbs are virtually identified together (“a tripod”), and driven synchronously, generating an alternating sequence of foot-falls to drive the chassis forward.

A celebrated control strategy responsible for achieving an alternating tripod gait, and enabling dynamic running, is the Buehler clock (see Fig. 3.11) ([Galloway et al., 2010](#); [Saranli et al., 2001](#)), which is a clocked (either by time, or a monotonic transformation of time referred to a “phase” in the literature) reference trajectory that is tracked by each motor. The Buehler clock gait was originally generated using a centralized agent commanding six motors with a time or phase domain reference tra-



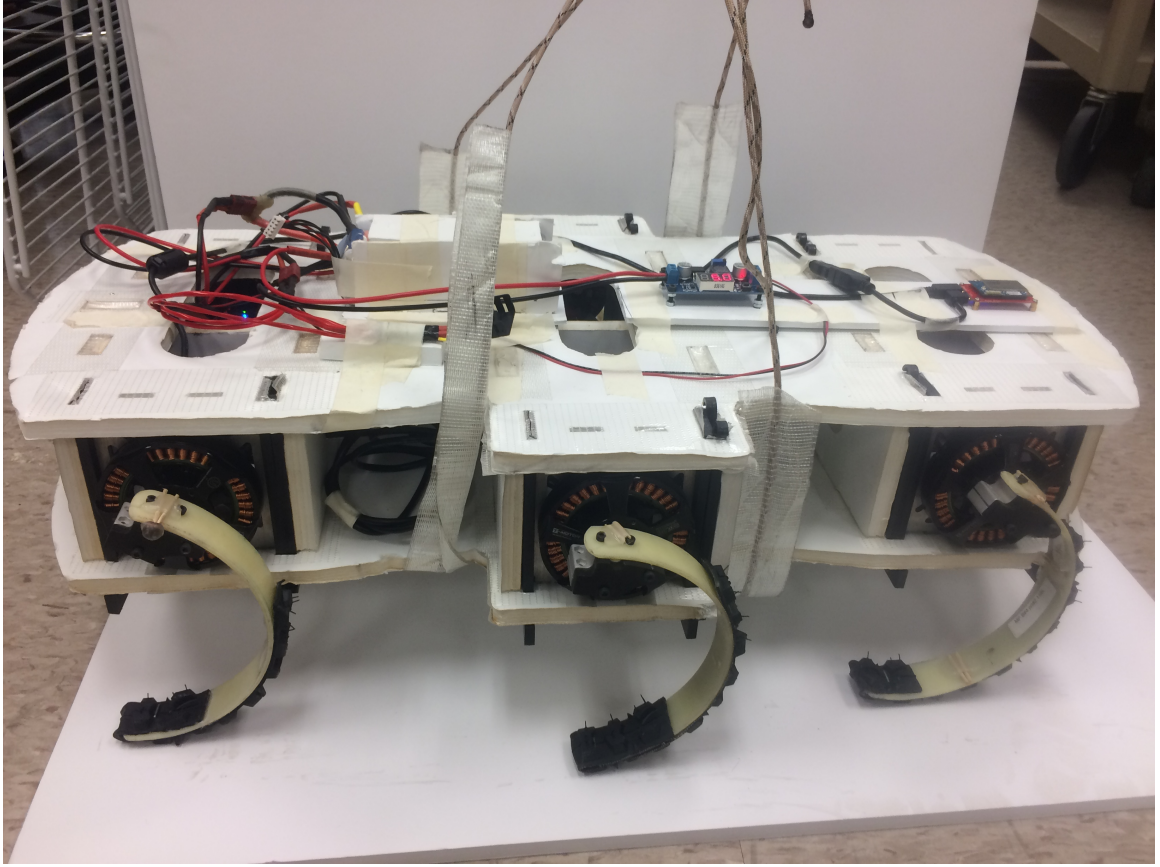


Figure 3.10: An image of our Rhex.

jectory driving the tripods either in a fast velocity or slow velocity. The slow region would intuitively correspond to the putative ground-contact phase, and the fast velocity applied would be during the aerial phase. In the ground contact phase, the legs are generating thrust, propelling the body forward, while the aerial phase recirculates the limbs to prepare for a subsequent thrust phase.

Classically, a central computer maintains periodic functions  $r_1(t), \dots, r_6(t)$ , which are passed to servomodules at times  $t_k$  for  $\delta k$  small compared to the frequency of  $r_i(t)$ . Each servo module instantaneously attempts to enforce command  $r_i(t_k)$  through a holding strategy until it is updated with command  $r_i(t_{k+1})$ . Each tripod of motors would receive identical reference signals, but  $T/2$  out-of-phase between tripods. (fig: 3.11).

Upon inspecting the graph in Fig. 3.11, we see that the Buehler clock is a polygo-

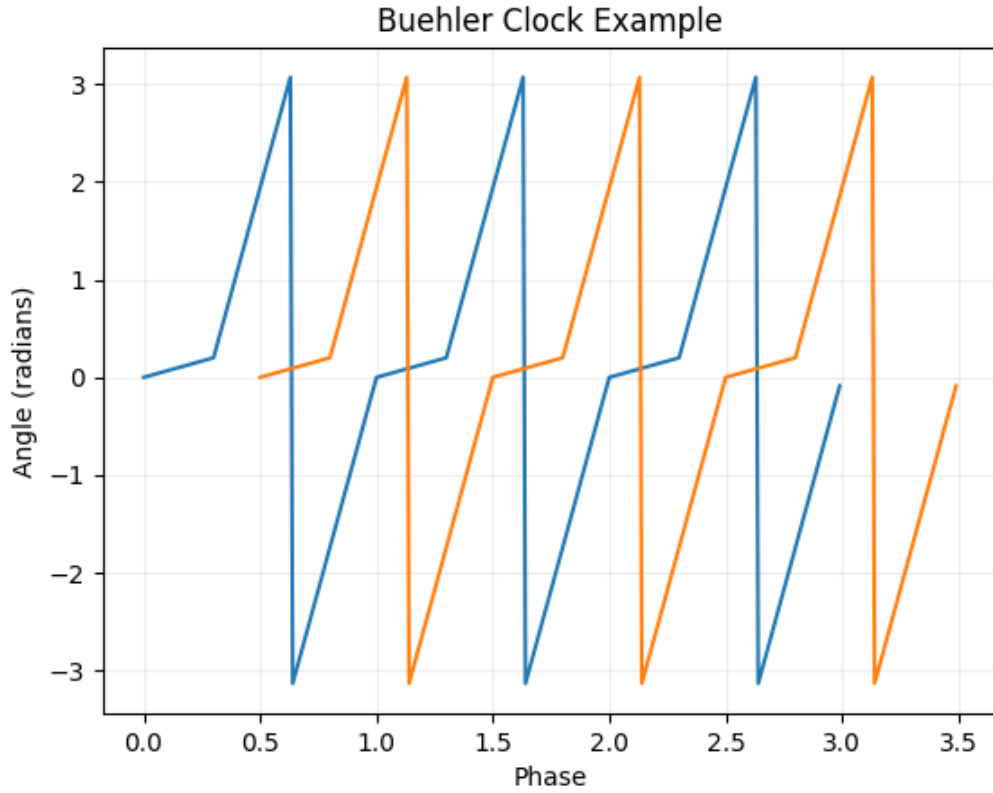


Figure 3.11: An illustration of a Buehler clock

nal curve, and that the knot points occur at putative event surfaces. We are motivated to define an event-selected vector field that produces a Buehler clock like curve as a limiting attractor. We expect comparable motive performance, with formal argument for being resilient against timing uncertainty and ground contact variation.

For an example  $EC^r$  field that has a Buehler-clock-like attractor, consult Fig. 3.13. A manually constructed event selected vector field is shown that has the periodic signal depicted in Fig. 3.12 as the limiting attractor. Event surfaces are denoted with the horizontal and vertical lines. Note that each surface is trivial, represented only by subspace crosses – detection of an event surface could be accomplished in software using only a sequence of if-then statements.

We make an important distinction regarding the scope of our approach. We claim that event selected vector fields are a sufficient mechanism to synthesize a gait

whose structure is inspired by a Buehler clock; i.e. an  $EC^r$  vector field is a novel type of controller to stabilize a desired periodic orbit in lieu of reference tracking via conventional closed-loop control. We do *not* claim that we know what the desired attractor should be – this is only attempting to stabilize a preferred choice.

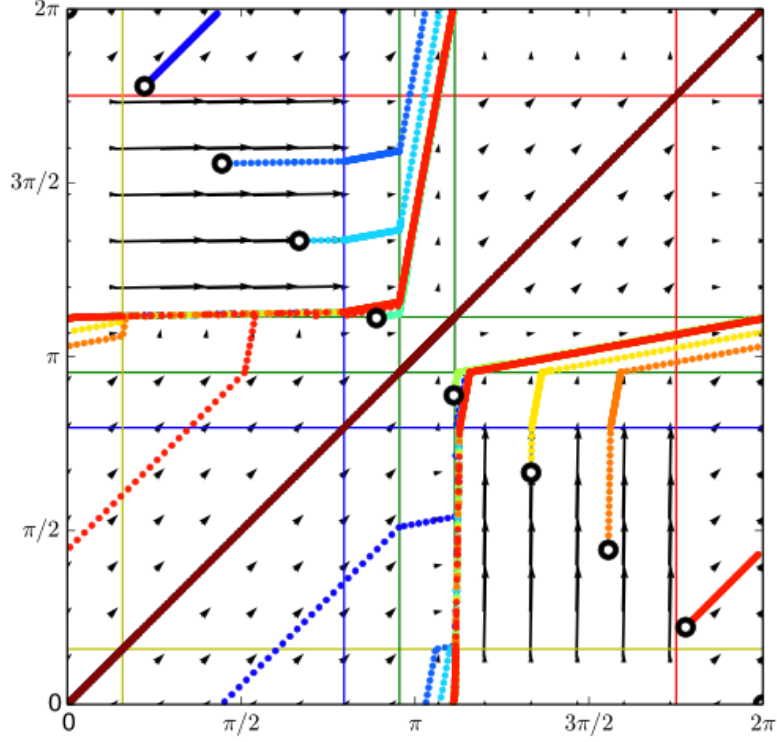


Figure 3.12: A two-dimensional ESS vector field on the torus. The off-diagonal dots are a collection of initial conditions that are mapped forward in time. The horizontal and vertical lines illustrate the event surfaces. Note that the green guard surfaces only change that coordinates' velocity component (e.g. horizontal green lines only change the horizontal component). These events, being only a function of one coordinate of the state, do not require global information. The other color of guards can change both components of the vector field. These indicate where a global event needs to be reported.

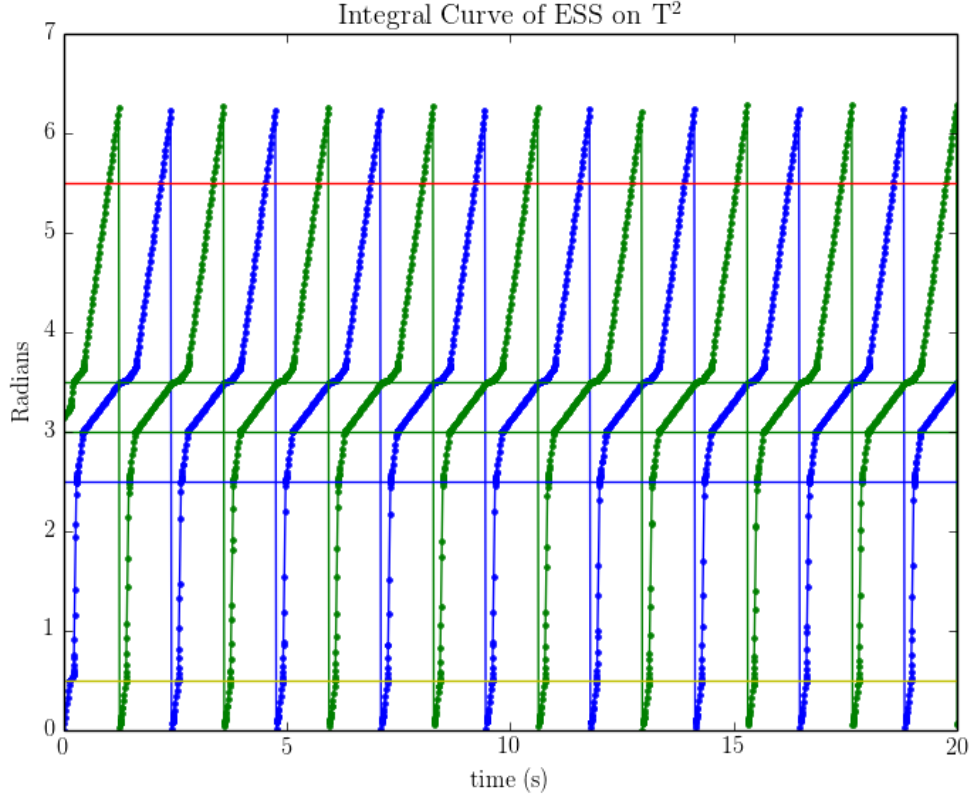


Figure 3.13: The periodic attractor plotted as a function of time. It features the analogous “fast-slow” behavior as a Buehler-clock. The horizontal lines indicate where the product guards (P-type) occur.

### 3.9.1 Piecewise Constant Vector Fields via Chambered First-Order Holds

For our use on RHex, we developed an implementation of  $EC^r$  control that enables the autonomous synchronization of legs that is parsimonious in both computation complexity and in communication bandwidth, but necessitates a servo with an expanded instruction set from those typically available.

Each servomodule is a joint microcontroller and motor as a monolithic collection that exposes a logical instruction set. They have a well-defined list of commands that can be accepted (such as command the motor shaft to a desired angle), and contain an internal PID control and commutation loop that energizes and commands the motors to achieve a commanded behavior. Using piecewise constant vector fields

allows RHex’s control action to be taken only at a finite number of event surfaces that discontinuously alter the governing vector field. Since we are interested in mandating this response from a collection of digital servomodules, we need to develop a language to represent a piecewise constant event selected vector field inside a computer.

We propose the network architecture is depicted as in Fig 3.14. Each distal node

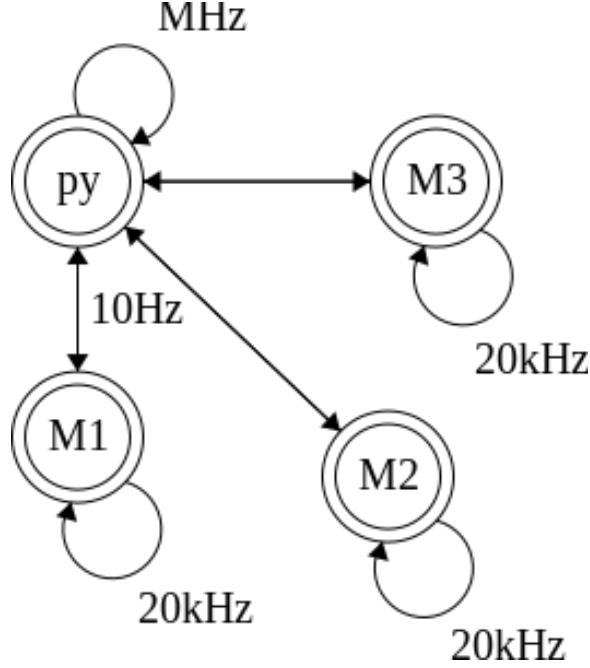


Figure 3.14: An illustration of the network topology and clock rates associated to RHex. M1,M2, and M3 are motor controllers, while “py” denotes the center CPU that interface with every controller. “10 Hz” is used to indicate low communication rates; since data is only irregularly sent per revolution, there is not an obvious way to make sense of the term “bandwidth”.

represents a motor controller – it contains the knowledge of its own state  $q_i$ , but does not know  $q_j$ ,  $\forall j \neq i$ . Since we are interested in piecewise-constant systems, we assume that each node has a controller capable of a first-order hold, i.e.  $q_i(t) = \dot{q}_{ref}t_{ref} + q_{ref}$  with reset. A node can be commanded to a specified position, and hold at a specified velocity.

The central node is responsible for supervisory planning, as well as the joint-state commands that result from crossing an exit section as described in §3.8. In a traditional model, the global information would be a sampled state  $q(k) = (q_1, \dots, q_n)(t_k)$ .

However, as we argued above, our telemetry is far more parsimonious. Furthermore, it is not even necessary that the central authority have explicit reports of every event crossing. A trajectory crossing the standard arrangement is a product condition – the value of  $q_i$  does not affect if  $q_j$  has crossed from one half-space into another. The sensing of exit surfaces can be reduced to products to eliminate the need for quantitative knowledge of a motor’s state, exactly as described above for stabilizing polygonal curves.

### 3.9.2 Software Architecture and Data Protocol

For our RHex, the central commanding node is a Linux kernel running a python process. It both provides a user interface for the six motor modules, and manages global state updates. The robustness of  $EC^r$  control mitigates the poor real-time performance of python.

Each servomodule accepts three fundamental types of commands: guarded, immediate, and preempts. Each node also contains  $n$  linear buffers  $B_i$  that can individually be read from and written to. There is also circular pre-buffer used to direct traffic to each linear buffer. A command appears as a prefix indicating its type, and a word carrying arguments. Arguments include a sub-type and its values. A command can have type “guard”, subtype “time”, “state”, etc, and value “1.2 seconds”, “2.1 radians”, etc. All incoming commands are typed via the pre-buffer, and processed by that type. The preempt commands are always immediately executed by the module as soon as they appear in the pre-buffer, and are for buffer management. They minimally set the active read buffer  $B_r$ , set the active write buffer  $B_w$ , manage the  $B_r$  buffer cursor, and purge the contents of a given buffer  $B_i$ . The guards and words are queued into the active write buffer  $B_w$  in the order they are received. Each node attempts to pop from the active read buffer  $B_r$ , and evaluate that command. Guards inhibit this pop based on their type and argument.

The guards are Boolean tests that form barriers in reading of the contents of the active read buffer. The processing of the buffer’s contents is not allowed to proceed until the condition associated to that guard is true (“clears the guard”). Once a guard is cleared, the contents of the buffer are read in order up until the next guard. The buffers  $B_i$  contain sequences of library commands separated by guards, allowing sequences to be chambered, where each block is controlled through guard conditions and selection of the appropriate buffer. The typical use-case is that the we pre-load the read buffers with behaviors at start-up. Each buffer contains the command sequence that defines a desired piecewise constant vector field. Then, the central authority need only send the set bit to select the right read buffer when global event surfaces are used, creating a very terse communication framework.

While the strategy requires little information exchange during an execution, we observe that each message is critically important. Lost packets violate the assumptions of the  $EC^r$  theory (transitions may not occur with lost packets), resulting in poor tracking. We therefore acknowledge that while we have the need for little data exchange, each datum is very information rich, necessitating the use of a protocol that enforces information parity.

For a complete specification, a Unified Modeling Language (UML) ([Rumbaugh et al., 2004](#)) sequence diagram is shown in Fig. 3.15.

To implement an  $EC^r$  vector field, guards are used to define event surfaces, with the subsequent post-guard messages ultimately being an affine position command  $q_i(t) = \dot{q}_{ref}(t - t_{guard}) + q_{ref}$  for the inner loop of the  $i$ -th servomodule. Internally, a servomodule tracks  $q_i(t)$  with high-gain PID, which is running at a much higher bandwidth than the communication channel between the central authority and the servomodule. Functionally, each servomodule is tracking a “primitive” with high-bandwidth local feedback, while the central authority stipulates the temporal organization of the primitives with performance guarantees arising from the ESS theory.

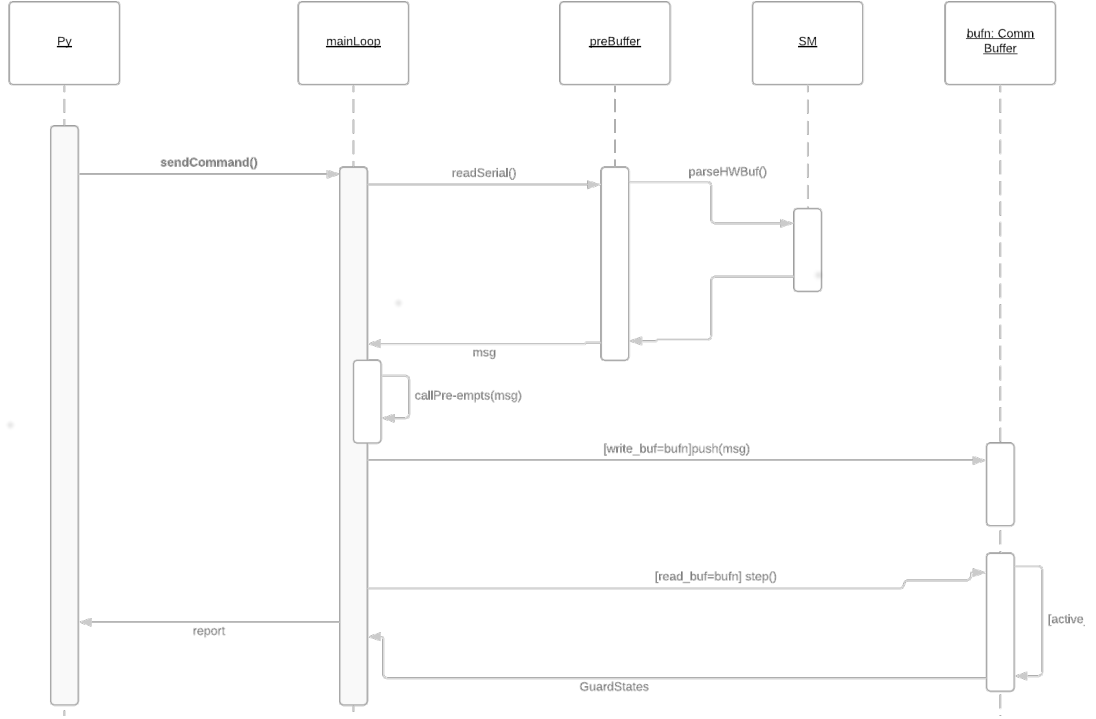


Figure 3.15: UML sequence diagram depicting the message flow sequencing as a function of time between the host CPU and a single motor. The commBuff classes are the linear buffers from above. SM is a state-machine that parses incoming data payloads into executable statements.

We now describe the types of guards implemented, as they both a) make concrete the above architectural discussion, and b) inform the class of possible behaviors, as there are non-equivalent guard regimes we expect to be of value.

Type	Symbol	Argument
Time	T	$t$ seconds
Position	P	$(\theta_1, \theta_2) \subset S^1$
Error	E	$e_{th} \in \mathbb{R}$
Contact	G	$b \in \{0, 1\}$

The first type of guard is a time guard. A time guard prevents execution until  $t$  seconds have elapsed since the last clock reset, i.e. time in the current epoch. Time guards allow us to create reference signals as a specification of time. For example, if the following sequence was in read buffer  $B_r$  for a position goal of  $p(t)$ .



Address	Symbol	Argument
0	T	0
1	Pos,vel	$(0,\pi)$
2	T	1
3	Pos,vel	$(\pi,2\pi)$
4	T	2
5	Clock	0
6	Cursor	0

The motor would advance to position zero and hold at  $\pi$  radians per second for one second. It would then move to  $\pi$  (in this case, the curve is continuous at  $\pi$ ) and hold at  $2\pi$  radians per second for 1 second. It then resets the clock and cursor to value zero, creating a periodic waveform with a period of two seconds. Here, the event surfaces are in time, allowing us to create a reference trajectory as a function of time. While this regime does is not suggestive of the advantages provided by an autonomous  $EC^r$  field, it allows us to program a traditional Buehler clock where the central authority needs to only synchronize the motor clocks, which is still lower in communication complexity than closed loop tracking.

The guard types more directly exploitative of  $EC^r$  properties are the P, E, and G type guards. The P-type guards are barriers in position. As we are operating on  $S^1$ , inequalities are not canonically defined, so implementing a “device” requires the topology of  $S^1$  be respected. Due to this, the position guards evaluate if the current state  $\theta_k \in (\theta_1, \theta_2)$  For small  $\delta\theta := \min(|\theta_1 - \theta_2|, |\theta_2 - \theta_1|)$ , the condition appears in practice as a single Boolean event, as the leg is presumably moving quickly. For  $\delta\theta$  large, we can denote a large section of  $S^1$ . This has utility for repeating commands (which can be a useful feature for communication protocols to avoid excessive acknowledgment replies), e.g. if a  $P$  command is sent twice, the condition will clear twice supposing the dwell time of the limb in the region is much longer than the

retransmission time. A practical method determining  $\theta \in (\theta_1, \theta_2)$  is given by Eqn. (3.32). Let  $\psi = \frac{\theta_1 - \theta_2}{2}$ ,  $\omega = \frac{\theta_2 - \theta_1}{2}$ . A representative example of this choice of guard function is depicted in Fig. 3.16.

$$\cos(\theta - \psi) \geq \cos(\omega) \quad (3.32)$$

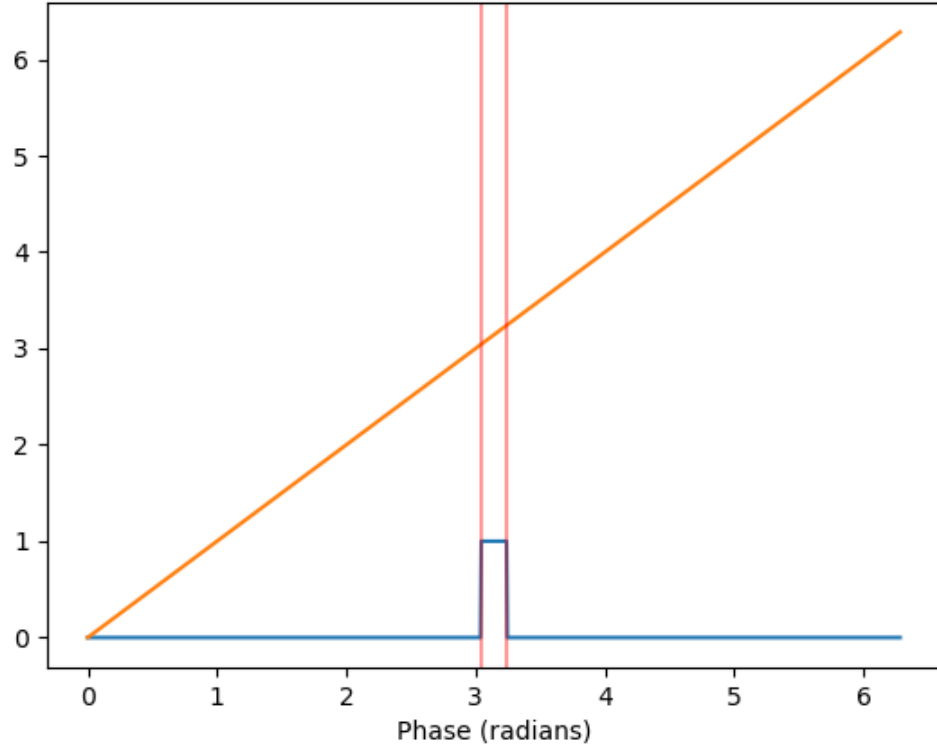


Figure 3.16: An example of Eqn. (3.32) as a Boolean decision.  $\theta_1 = \pi - 0.1$ ,  $\theta_2 = \pi + 0.1$ , the window of which is indicated by the red vertical angles. The orange curve is the phase curve  $\theta(t) = t$ , while the blue curve is the Guard value - when  $\theta \in (\theta_1, \theta_2)$ , the function returns true.

As a main example, the “device” of 3.7 could be implemented on two servomodules using velocity commands (“vel” type with “arg” value, which sets  $\dot{\theta}_{ref} = \arg$ ) commands (e.g. with position gain 0) by populating their buffers identically as:

Address	Symbol	Argument
0	Clock	0
1	vel	1
2	P	$(\pi, 2\pi)$
3	vel	$1-\delta$
5	P	$(\frac{3\pi}{4}, 2\pi)$
6	Report	1

The “Report” commands are used to identify crossing the exit section in the upper right quadrant of the square torus, i.e.  $q_1 \in (\frac{3\pi}{4}, 2\pi)$ ,  $q_2 \in (\frac{3\pi}{4}, 2\pi)$ . The central authority monitors for both reports - if their logic AND is true, it simultaneously resets the cursors back to 0.

The “P” type guards are functions of angle, so define an  $EC^r$  field on  $T^6$ . However, we wish to have the capacity to have hybrid events occur at physical ground contact, rather than a supposed contact. Thus, we need additional guard types.

The next type is an error guard. Error guards evaluate the tracking error  $e(t) := q(t) - q_{ref}(t)$  used in the servomodules inner loop controller. If  $\|e(t)\| > e_{th}$ , the guard clears.

In light of the Buehler-clock specification we are attempting to achieve, where a “fast” aerial speed precedes a “slow” ground speed, consider the following command sequence.

Address	Symbol	Argument
0	Clock	0
1	vel	1
2	E	$(\pi, 2\pi)$
3	vel	$1-\delta$
5	P	$(\frac{3\pi}{4}, 2\pi)$
6	Report	1

If the tracking controller is performing well (we assume that it is), the error guard will not clear until the limb impacts the ground, where the leg is forced to slow down, increasing error.

We observe that the behavior of the robot under this scheme is distinct if the robot were lifted clear of the ground versus being in contact with the ground. We would simply see the legs spin at constant velocity if the robot is airborne. The interaction of the limb with the ground is *essential* to the genesis of a stable periodic gait. We admit an imperfect understanding of the performance expected by such a strategy – accurate detection of the ground contact event essentially relies on selecting the right error threshold value. Finding such a value would be incumbent upon user tuning, and would likely depend strongly on the regularity of the ground, and the choice of PID tracking gains.

The final type of guard our modules implement guard condition on external sensor data, which is indicated with a type of “G”. The argument of this guard type is a placeholder – the sensor reading is assumed to be a digital pulse that aligns with ground contact. When a ground contact even occurs, the sensor switches from “0”, to “1”. A servomodule’s main loop periodically samples this signal (typically at rates  $\geq 20\text{kHz}$ ). If it is “1”, the guard clears, and the buffer address is advanced. Our implementation uses a simple contact switch mounted on the bottom of the RHex legs connected via photocoupler to the CPU. Additional details will be provided in §3.9.4 section below.

Currently, we only have ad-hoc tools to compose command sequences to stabilize desired polygonal curves. The proposed polygonal curve algorithm above is only a partial answer, as it is formulated to specify only a local  $EC^r$  field. In the sequel we will present a simulation framework designed to facilitate the development of these command sequences by providing an environment in which sequences can be tested via trial-and-error.

### 3.9.3 Simulation

Running physical experiments on the robot is time intensive, and prone to damaging the device when a non-viable gait is tested. To immunize our project against this, we have developed a pure-software python simulation to validate command sequences for the RHex robot. To emphasize, it is a simulation of the network of modules, not the dynamics of RHex. It uses the exact interface used to drive the physical modules mounted on RHex with virtual, emulated versions that implement the protocol described above. Doing so allows us to determine if an undesirable gait is a result of the poor usage of the intended logic of the control scheme, or from bugs or other limitations in the hardware. The python simulation implements error-less transmission – the commands sent by the central authority are simply deposited in the buffers of the virtual modules without modeling packet loss.

Beyond implementing the control protocol, the virtual modules include several non-ideal factors to test the resiliency of the gait to plausible or inescapable features of the physical hardware. We implemented (with configurable parameters):

- Command tracking bandwidth and noise. A reference command response of the first-order hold includes a first-order low-pass model of the motor to represent the inability of a physical device to move arbitrarily quickly to a desired location, simulating the feedback loop of the servo.
- Finite-difference velocity estimation, where sampled data from the above low-pass model is corrupted with additive or multiplicative noise drawn from either bounded uniform or normal distribution, indicating that sensing is imperfect, or that unknown dynamics disrupt the velocity and position signals.
- A sampling outbox with packet loss. A memory structure is populated by a down-sampled version of the virtual modules own state. It includes irregular updating, and packet loss. We use this feature to represent that the central

authority can only query a module’s state at some sampling frequency over an unreliable channel.

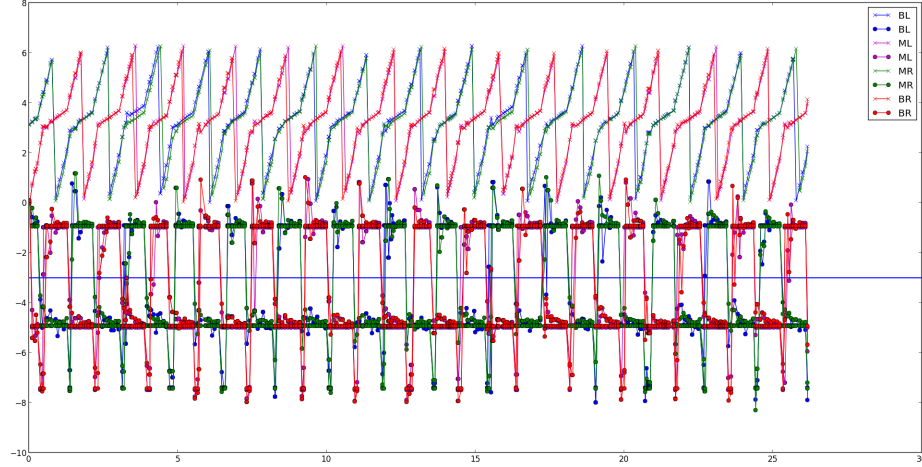


Figure 3.17: A simulation output in the time domain of the virtual RHex. The blue and green triangular waveforms are the positions of each leg. The controller is grouping them well into two tripods. The lower square-like waves are the corresponding velocities. The jaggedness is a result of injected noise and model irregularities that are being rejected via the event-selected feedback.

Our simulation framework allows us to determine if a programmed behavior is going to be persistent to tracking quality and encoder noise. While structural properties of the flow of an  $EC^r$  flow guarantee  $C^0$  closeness under perturbation, we must additionally account for difficulties arising from computation time or quantization error, e.g. if the dwell time for a type “P” guard is lower than the sampling frequency of the servomodule controller, it fails to properly detect the guard. As the simulation uses the same software interface as the physical robot, the code we use to test the virtual modules is *directly* used to drive the hardware – merely a flag “hw=True” is set, and the exact same code is executed. The hardware abstraction layer running beneath the interface is responsible for managing the differences between the hardware and virtual servomodules. By including these features into our software, we hope to reduce development time.

Shown below in Fig. 3.17 is an example output of this simulation, where we are synthesizing a Buehler clock across six modules in an alternating tripod gait. We see that despite the addition of disruptive factors, our desired signal remains strongly synchronized and periodic.

### 3.9.4 Hardware

Due to the non-standard control protocol proposed above, we found it necessary to develop custom servomodules, as no commercial driver had the ability to implement our guard-based buffer scheme. We collaborated with Ghost Robotics, and modified a donated servomodule to implement the protocol described in §3.9.2. We modified both the firmware and circuitry. Additionally, we required both ground contact sensors and power management for the robot, and designed in-house custom options for this task.

As mentioned, the motors are a custom evolution of a Ghost Robotics prototype that incorporates commutation, communication, and servo-control into a single platform. The motors are commercially available 21 pole-pair, U8 KV100-series motors produced by Tiger Motor in direct-drive configuration with the fiberglass leg. The motivation for direct-drive was increased transparency of ground contact in error dynamics, avoiding damage to gear boxes through impact, and other factors ([Kenneally et al., 2016](#)).

#### 3.9.4.1 Ground Contact Sensing

We mounted contact switches to bottom of the RHex legs to sense ground contacts. Since the legs undergo complete rotations, we elected to use an optical coupler to connect the switching signal to the microcontroller. While a slip-ring may be a more conventional choice for such a task, the mechanical limitations of our design did not facilitate the easy inclusion of a slip ring. Additionally, slip rings have intermittent mechanical contacts, leading to both wear and debouncing complications, neither of

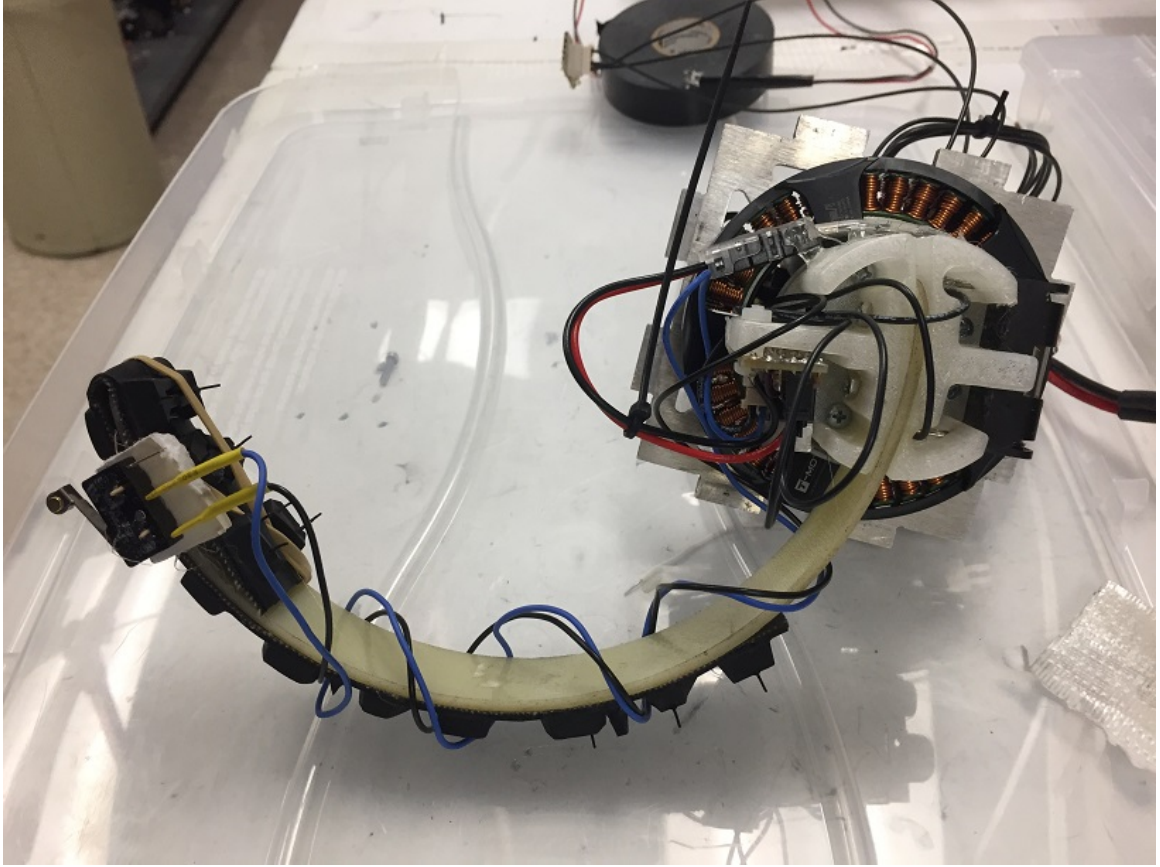


Figure 3.18: A complete RHex leg. The fiber-glass leg is load-bearing, while it attaches to the BL-DC motor driver via a 3D printed fiber-reinforced nylon mount. The motor is bolted to the aluminum plate, which then attaches to the RHex frame. The visible wiring and electrical components are part of the optically coupled ground contact sensor. On the reverse, not pictured, is the motor controller and thermal safety circuitry.

which are problems with optical coupling.

Since we are interested in the ground impact itself, rather than an entire stance phase, we used the contact switch to drive a pulse-generator that illuminated a strip of LED's wound around the plastic leg mount. The circuitry and power supply were fully externalized and mounted on the leg, eliminating any wiring between the limbs and servomodules.

Since we wanted to detect a contact at any angle, we elected to use flexible LED strips wound entirely around the motor. A photodetector and amplifier were mounted on the RHex housing, which then directly connected to the microcontroller.



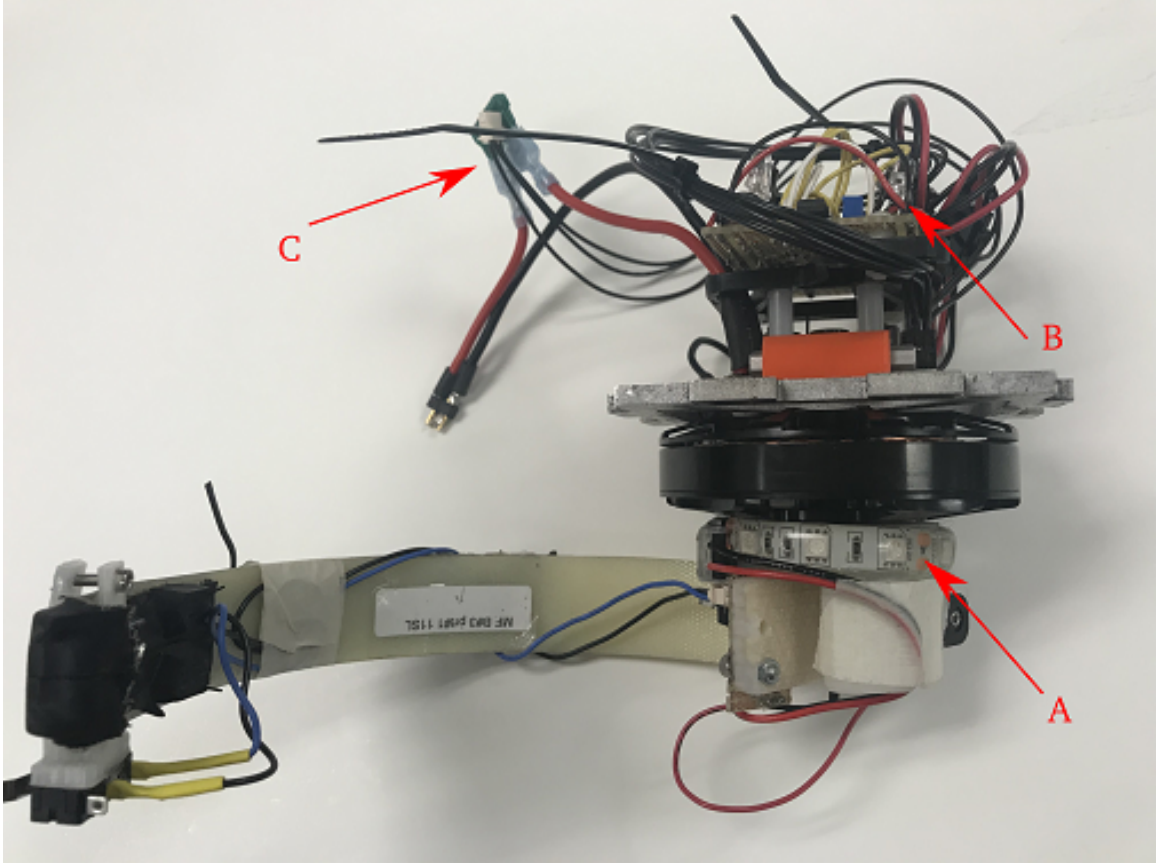


Figure 3.19: An above view of the servo-module and RHex leg. (A) is the flexible LED strip wound around the motor mount, (B) is the thermal safety daughter board, while (C) is the ATC blade fuse.

An example circuit that produces such an effect is depicted in Fig. 3.20.

#### 3.9.4.2 Thermal Safety

Given the large currents needed to drive the motors, thermal runaway is a possibility, especially during stall conditions. To mitigate thermal overload, we incorporated thermal safeties. The high-level function of the safety is to block traffic emitted from the central node that is directed at the overheating servomodule. Each servomodule expects a periodic heartbeat signal; if the signal is not received after a sufficient time gap, the module ceases to energize the motor, driving high currents to zero. The thermal safety is a collection of three Positive Temperature Coefficient (PTC) thermistors, thermally bonded to the surface of the H-bridge, that undergo exponential

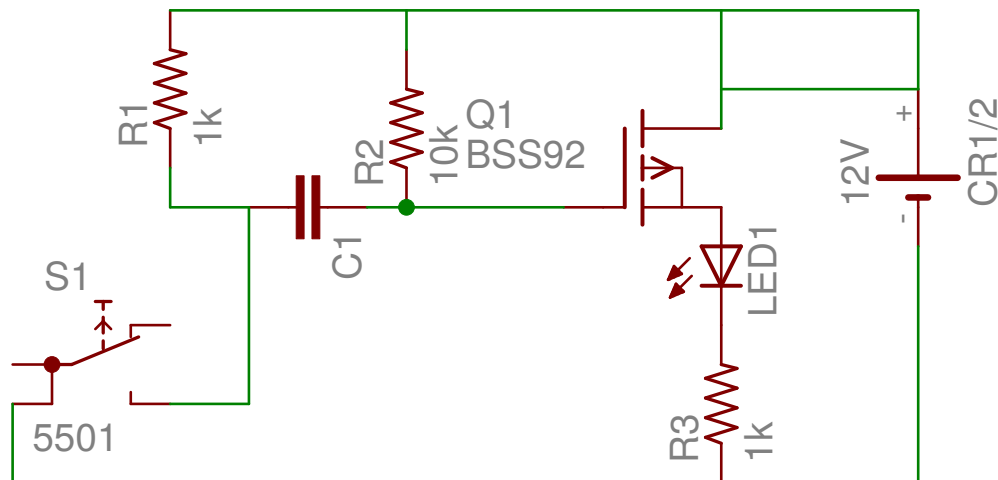


Figure 3.20: A simple pulse generator – the push-button closure illuminates LED1 with a bandwidth controller via R1,R2, and C1.

growth in resistance at a known temperate. With our component choice, the thermistors resistance jumps from 100 ohms to over 100 kilohms at an exponential rate around the set point of 90 Celsius. This discrete jump latches the enable pin of a tri-state buffer that is in series on the serial transmission line. When the buffer is tripped, it presents as high-impedance, terminating traffic. Three such buffers (one for each H-bridge) are sequentially connected, so that a thermal overload in any of the H-bridges generates a fault condition. The schematic for this device is shown in Fig. 3.23.

### 3.9.4.3 Power Regulation

The RHex is designed to be driven by a 4S1P Lithium-Polymer (LiPo) battery operating nominally between 13.5 V and 16.6 V. Since the computational components of the motor modules are driven from a USB2.0 port, their total draw is limited to  $6 \times 500 \text{ mA} = 3\text{A}$ .

Ergo, the motors are the dominant current loads. Each H-bridge is populated with



Figure 3.21: An example trace of the resulting input signal to the microcontroller. The green trace is the signal produced by switch, while the yellow trace is the resulting output of the photo-edge detector.

three BTN8980 half-bridge driver chips, which can tolerate up to 44 A per phase line, with surges up to 110A for 10 ms. Collectively, six motors in parallel can draw in excess of 600A for short a duration.

However, this is a very high performance regime. For the initial prototyping, currents per motor (sum of all three phase currents) of 15A are sufficient. Since LiPo batteries are easily capable of producing currents in the hundreds of amps, we need to protect against high-current surges. We elected to simply incorporate 15A ATO LittleFuse 32 V blade fuses ([ato, 2019](#)) into the hot line of each motor module. Each LTC fuse is “slow-blow”, so that motor transients below 200% of 15A can be tolerated for 5 seconds before the fuse blows.

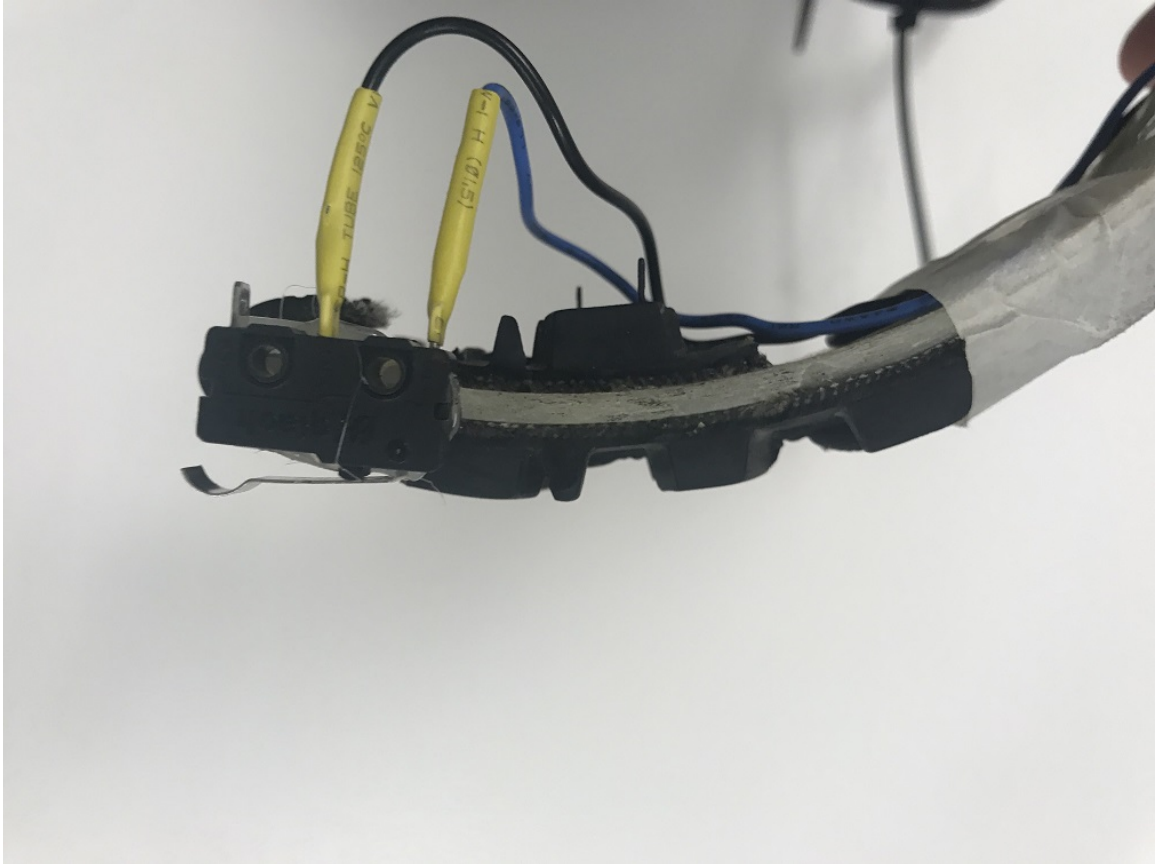


Figure 3.22: The contact switch used to indicate if the leg has impacted the ground. When the beam is depressed by contact, the receiving circuitry converts this event into a 50 ms pulse.

We additionally require undervoltage lockout. If a LiPo battery is sufficiently discharged, it can catastrophically fail. Given the high currents tolerated by the RHex motors, a 50C 4000 mAh battery, our standard power source, can be depleted in a matter of minutes. The protection circuit is designed to be latching, so that if the undervoltage condition is ever achieved, the power outputs are disabled until the circuit is de-energized for maximum safety.

Finally, as the motor drivers possess a significant electrical reactance, in-rush currents can be substantial, which can cause destruction failures. We must protect against large in-rush currents until the time-constants of reactive components have elapsed, and their current draw dissipates.

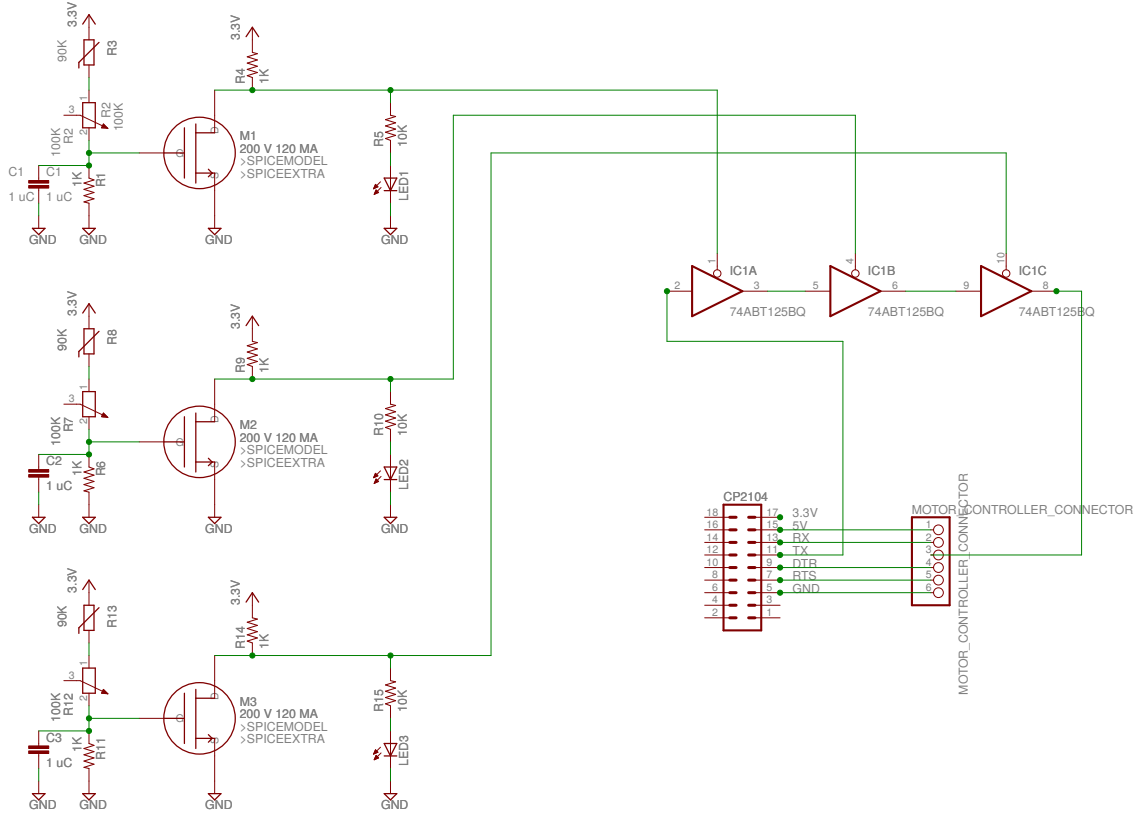


Figure 3.23: The thermal safety schematic. There are three channels , one for each branch of the H-bridge, that are multiplexed together to TX line of the python kernel. Suppression of traffic on this line violates the heartbeat timer, resulting in a shutdown of the servomodule.

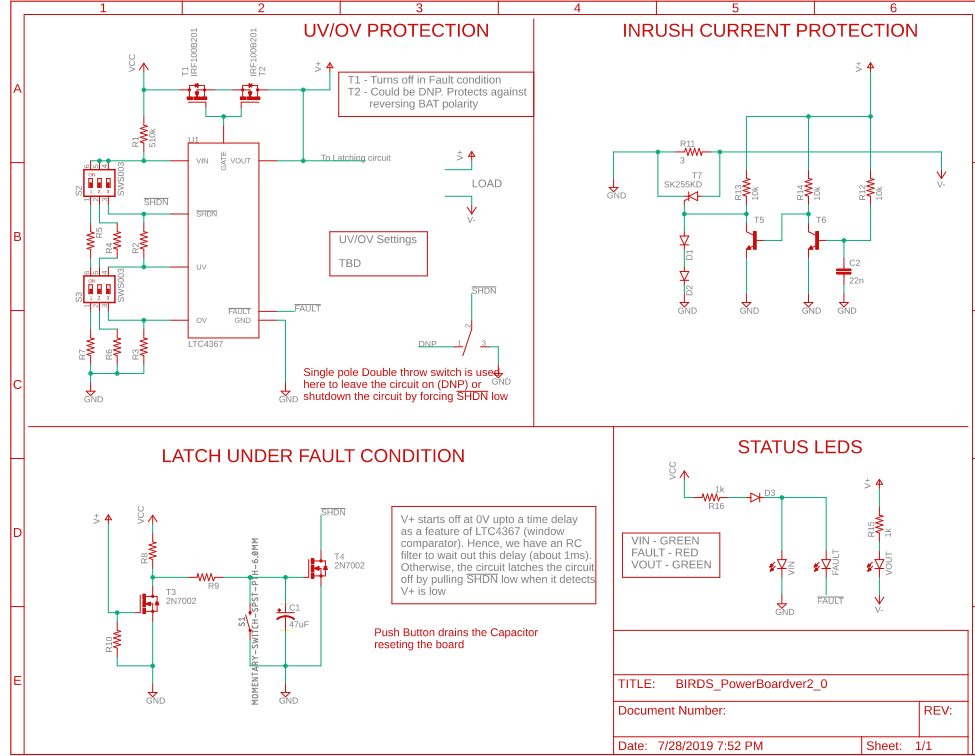
Shown below in Fig. 3.26 is a schematic that accomplishes our desired goals.

### 3.9.5 Performance

#### 3.9.5.1 Reference Tracking

Our custom protocol allows the implementation of a standard Buehler clock with reference tracking through the usage of time guards. As a sanity check, we elected to verify that our method could produce an alternating tripod gait.

Such a gait produces stable forward translation of the RHex chassis, as we would expect. The gait is “walking”, wherein it is statically stable, or nearly so, at all points during its stride. A representative trace of the CoM while using this gait is displayed



10/21/2019 11:12 PM f=0.90 E:\BIRDS\power board\ver2.0\save5\BIRDS\_PowerBoardver2\_0.sch (Sheet: 1/1)

Figure 3.24: Power module schematic. The function in each section is described in text.

in Fig. 3.27. Two motion capture markers were attached to the upper surface of RHex to define a course COM, and filmed with the Qualisys motion capture system at 60 FPS on five Opus 310-series cameras. The robot walked directly ahead, so the  $y$ -coordinate motion was relatively constant. The  $x$ -coordinate roughly linearly increased. Our purpose of this section is to indicate that we successfully implemented reference tracking in our custom RHex, which is the conventional method of control.

### 3.9.5.2 Piecewise-Constant Performance

The performance of the  $EC^r$  distributed code on hardware was so far not yielded a reliable data set.

While the novel protocol from above is implemented, we have encountered difficulties in reliable communication between the servomodules and the central python

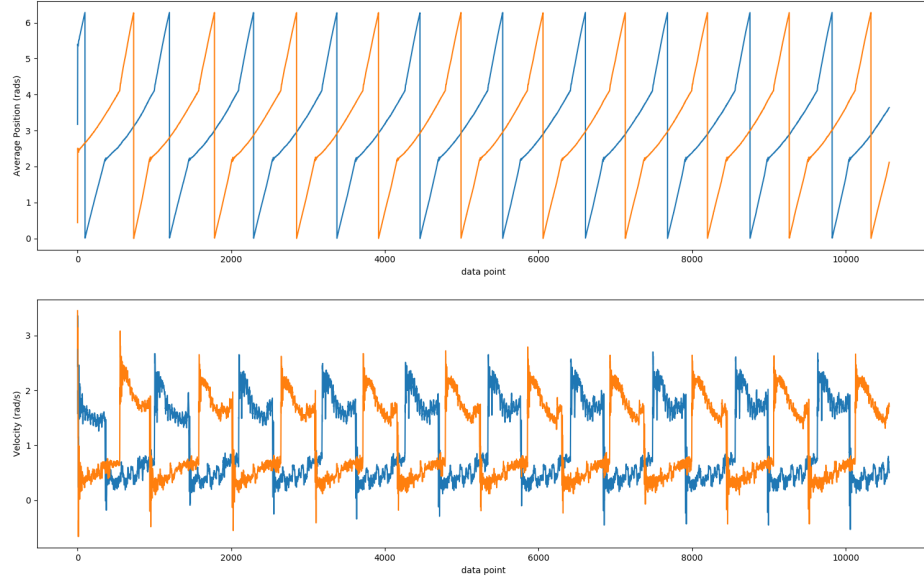


Figure 3.25: Reference tracking an alternating tripod gait in dry-dock, where the legs do not make contact with the ground. The upper plot is the average angle of each tripod, while the lower figure is the velocity.

authority.

In Fig. 3.28 and Fig. 3.29, we display the current state of progress. The physical telemetry is reported by the servo modules as an instrument to validate performance – it is not used for control actions. We see that there seems to be reasonable agreement between the simulated results and the motion of the robot. Data was collected in dry-dock (the robot is suspended in the air), free of unexpected contacts – ergo, the guards were of “P” type. It is also salient that the polygonal curve is not a meaningful gait. It is an arbitrary trial behavior intended to validate the implementation.

We admit there remains work to be done is demonstrating that a fully distributed control scheme can produce a reliable gait, especially when ground contact sensors are used.



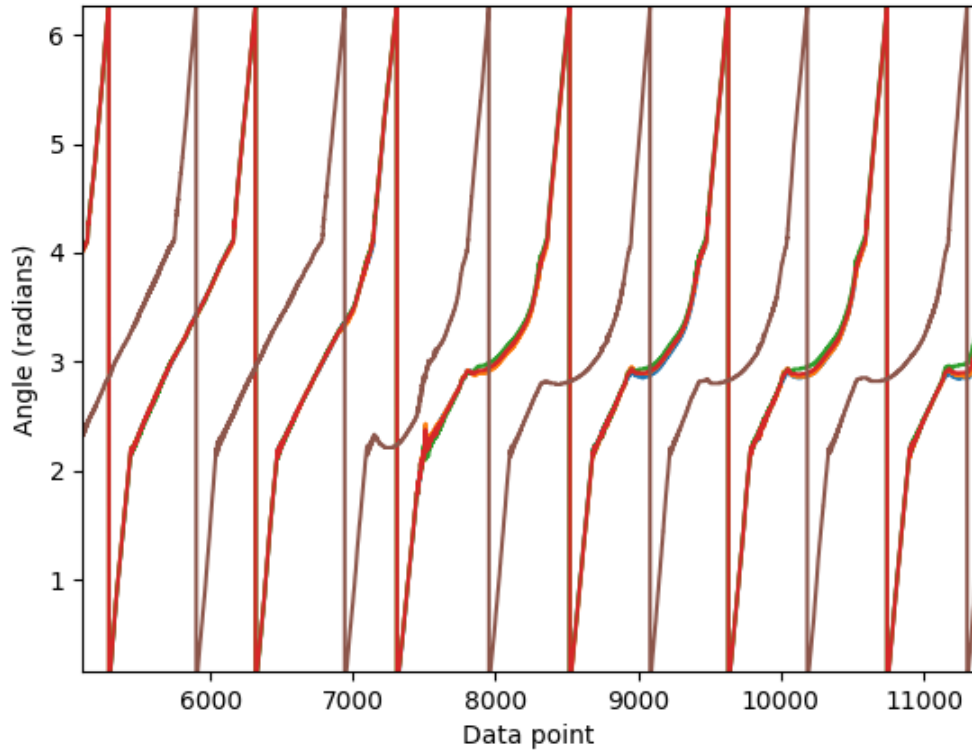


Figure 3.26: Reference Buehler tracking. The robot starts suspended in the air, and is then placed on the ground around data point 7000 to stride forwards. The high tracking quality degrades, as the legs in contact with the ground appear to experience a pure acceleration until their eventual liftoff.

### 3.10 Future Work

Two major directions immediately present themselves to continue the line of reasoning opened in this chapter. The first and most apparent is to continue the development of the RHex robot and resolve the underlying hardware issues that inhibit the genesis of a useful gait with event selected control. In particular, using the ground contact events as guards to synthesize a stable gait would be a novel departure from conventional autonomous legged controllers. It would provide an demonstration that periodic gaits do not need to be exclusively generated from tracking periodic reference trajectories, but can rather arise from predictable interaction with the environment.



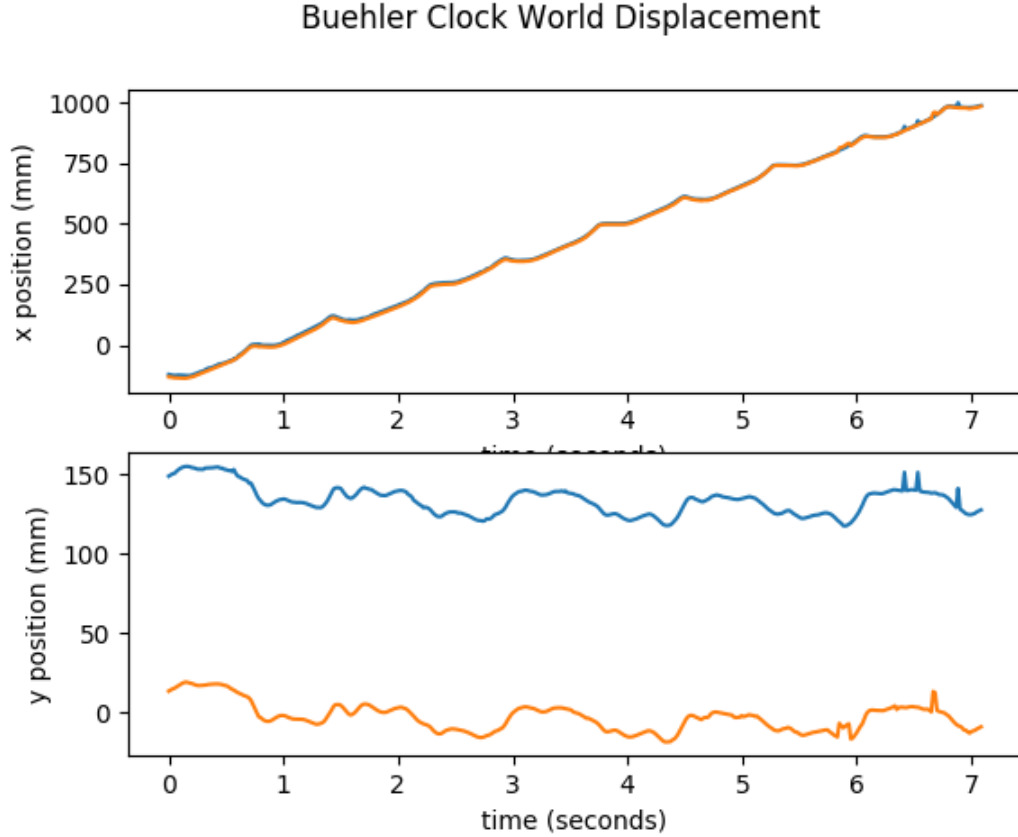


Figure 3.27: Reference Buehler tracking. The blue trace is one marker, while the orange trace is the other. The upper plot is the  $x$  coordinate of the plane, while the lower plot is the  $y$  coordinate of the plane. Since the robot is not airborne, the  $z$  coordinate is omitted.

Secondly, the numerical estimation of the linearization of a  $PC^r$  has apparent application in multi-contact locomotion and manipulation problems. The validation examples we used in §3.3 were quite artificial. The algorithm could potentially be used to dramatically accelerate multiple contact planning problems (*Tassa et al.*, 2012; *Mordatch et al.*, 2012a) as we can directly account for the effect of variations *through* contact without resorting to a combinatorial number of integrations.

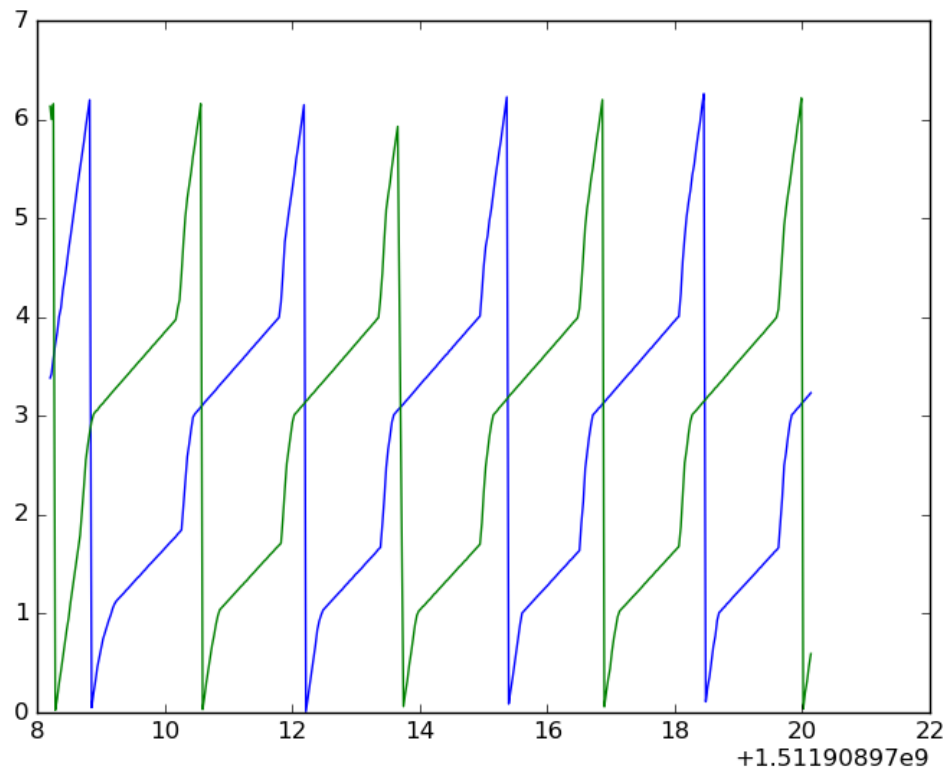


Figure 3.28: Simulated trajectory data down-sampled via the host controller using P-type guards. Zero noise was added to the response. Finite-response time was enabled, smoothing the knot points.

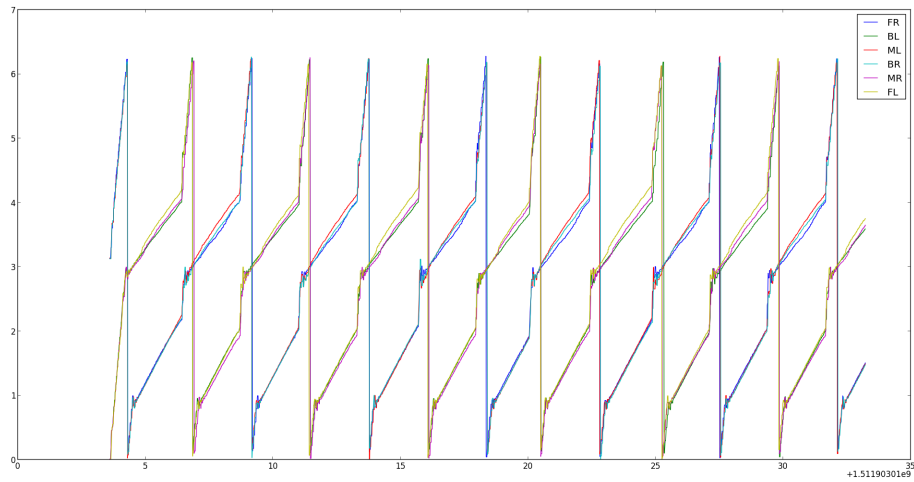


Figure 3.29: The physical robot executing the same command sequence – while irregularities emerge, it is clear the same qualitative structure as the simulated response is present.

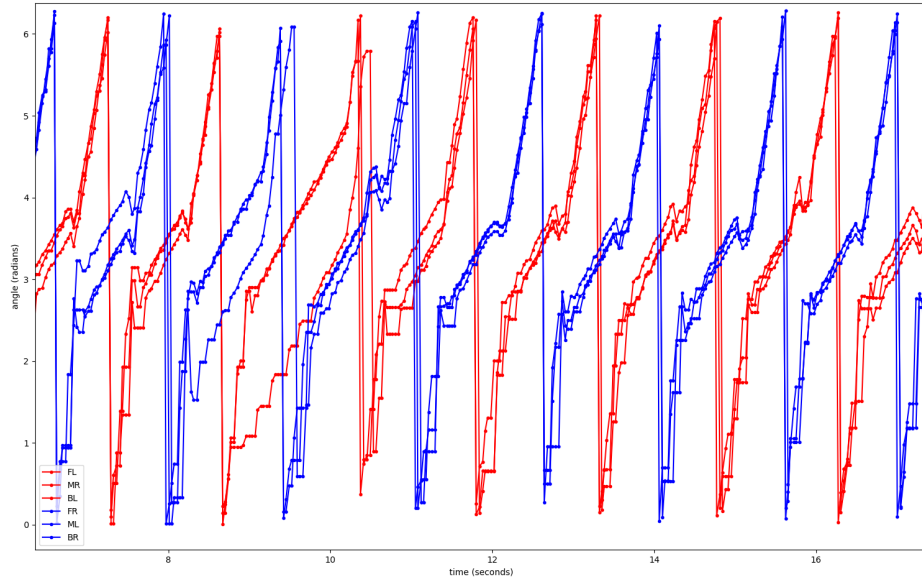


Figure 3.30: A mixed regime controller. The in-tripod synchronization is accomplished with distributed event-selected control, while the phase difference between the tripods is done with feedback. Red traces are one tripod, while blue traces are the other. We can see that a disruption to limbs is effectively rejected completely for in two strides. This data was collected in dry-dock.

## BIBLIOGRAPHY

### Bibliography

(2019), *ATOF Blade Fuses Rated 32*, LittleFuse.

Adolph, K. E., and S. R. Robinson (2013), The road to walking: What learning to walk tells us about development, *Oxford handbook of developmental psychology*, 1, 403–443.

Aizerman, M., and F. Gantmacher (1958), Determination of stability by linear approximation of a periodic solution of a system of differential equations with discontinuous right-hand sides, *The Quarterly Journal of Mechanics and Applied Mathematics*, 11(4), 385–398.

Alexander, K., et al. (1994), *Smooth invariant manifolds and normal forms*, vol. 7, World Scientific.

Alexander, R. M. (1984), The gaits of bipedal and quadrupedal animals, *The International Journal of Robotics Research*, 3(2), 49–59.

Ankarali, M., and U. Saranli (2010), Stride-tostride energy regulation for robust self-stability of torque-actuated dissipative spring-mass hopper, *Chaos*, 20(033121), doi:10.1063/1.3486803.

Arnold, V. (1973), *Ordinary differential equations*, The MIT Press, Cambridge, Massachusetts.

Arslan, O., and U. Saranli (2012), Reactive planning and control of planar spring-mass running on rough terrain, *IEEE Trans. Robotics*, 28, 567–579, doi:10.1109/TRO.2011.2178134.

Bellman, R. E. (1961), Dynamic programming treatment of the traveling salesman problem.

- Bernardo, M., C. Budd, A. R. Champneys, and P. Kowalczyk (2008), *Piecewise-smooth dynamical systems: theory and applications*, vol. 163, Springer Science & Business Media.
- Bizzarri, F., A. Brambilla, and G. S. Gajani (2013), Lyapunov exponents computation for hybrid neurons, *Journal of computational neuroscience*, *35*(2), 201–212.
- Blickhan, R. (1989a), The spring-mass model for running and hopping, *J. Biomechanics*, *22*(11-12), 1217–1227, doi:10.1016/0021-9290(89)90224-8.
- Blickhan, R. (1989b), The spring-mass model for running and hopping, *J. Biomechanics*, *22*(11-12), 1217–1227, doi:10.1016/0021-9290(89)90224-8.
- Bloch, A., J. Baillieul, P. Crouch, J. E. Marsden, D. Zenkov, P. S. Krishnaprasad, and R. M. Murray (2003), *Nonholonomic mechanics and control*, vol. 24, Springer.
- Bloch, A. M., P. Krishnaprasad, J. E. Marsden, and R. M. Murray (1996), Nonholonomic mechanical systems with symmetry, *Archive for Rational Mechanics and Analysis*, *136*(1), 21–99.
- Bogert, A., K. Gerritsen, and G. Cole (1998), Human musclemodelling from a user’s perspective, *J. Elecromyogr. Kines.*, *8*(2), 119–124, doi:10.1016/S1050-6411(97)00028-X.
- Bongard, J., V. Zykov, and H. Lipson (2006), Resilient machines through continuous self-modeling, *Science*, *314*(5802), doi:10.1126/science.1133687.
- Bongard, J. C. (2011), Morphological and environmental scaffolding synergize when evolving robot controllers: artificial life/robotics/evolvable hardware, in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pp. 179–186, ACM.
- Browder, F. E. (1954), Covering spaces, fibre spaces, and local homeomorphisms, *Duke Math. J.*, *21*(2), 329–336, doi:10.1215/S0012-7094-54-02132-8.
- Bullo, F., A. D. Lewis, and K. M. Lynch (2002), Controllable kinematic reductions for mechanical systems: concepts, computational tools, and examples, in *Mathematical Theory of Networks and Systems*, vol. 124.
- Burden, S. A., S. Revzen, and S. S. Sastry (2015), Model reduction near periodic orbits of hybrid dynamical systems, *IEEE Transactions on Automatic Control*, *60*(10), 2626–2639.
- Burden, S. A., S. S. Sastry, D. E. Koditschek, and S. Revzen (2016), Event-selected vector field discontinuities yield piecewise-differentiable flows, *SIAM Journal on Applied Dynamical Systems*, *15*(2), 1227–1267.
- Carver, S., N. Cowan, and J. Guckenheimer (2009), Lateral stability of the spring-mass model suggests a two-step control strategy for running, *CHAOS*, *19*, doi:10.1063/1.3127577.

- Council, G., S. Yang, and S. Revzen (2014), Deadbeat control with (almost) no sensing in a hybrid model of legged locomotion, in *Advanced Mechatronic Systems (ICAMEchS), 2014 International Conference on*, pp. 475–480, IEEE.
- Cully, A., J. Clune, D. Tarapore, and J. Mouret (2014), Robots that can adapt like animals, *Nature*, *521*(7553), doi:10.1038/nature14422.
- Da, X., O. Harib, R. Hartley, B. Griffin, and J. W. Grizzle (2016), From 2d design of underactuated bipedal gaits to 3d implementation: Walking with speed tracking, *IEEE Access*, *4*, 3469–3478.
- Deisenroth, M. P., G. Neumann, J. Peters, et al. (2013), A survey on policy search for robotics, *Foundations and Trends® in Robotics*, *2*(1–2), 1–142.
- Dieci, L., and L. Lopez (2011), Fundamental matrix solutions of piecewise smooth differential systems, *Mathematics and Computers in Simulation*, *81*(5), 932–953.
- Ernst, M., H. Geyer, and R. Blickhan (2011), Spring-legged locomotion on uneven ground: A control approach to keep the running speed constant, in *Int. Conf. on Climbing and Walking Robots*, pp. 639–644.
- Farley, C., R. Blickhan, and C. Taylor (1985), Mechanics of human hopping : model and experiments, *Am. Zool.*, *25*, 54A.
- Fenichel, N. (1974), Asymptotic stability with rate conditions for dynamical systems, *Bulletin of the American Mathematical Society*, *80*(2), 346–349.
- Fenichel, N. (1977), Asymptotic stability with rate conditions, ii, *Indiana University Mathematics Journal*, *26*(1), 81–93.
- Filippov, A. F. (2013), *Differential equations with discontinuous righthand sides: control systems*, vol. 18, Springer Science & Business Media.
- Full, R. J., and D. E. Koditschek (1999), Templates and anchors: neuromechanical hypotheses of legged locomotion on land, *Journal of experimental biology*, *202*(23), 3325–3332.
- Galewski, M., and M. Koniorczyk (2016), *Global invertibility theorems and their applications-a variational approach*, Wydawnictwo Politechnika Łódzka.
- Galloway, K. C., G. C. Haynes, B. D. Ilhan, A. M. Johnson, R. Knopf, G. Lynch, B. Plotnick, M. White, and D. E. Koditschek (2010), X-rhex: A highly mobile hexapedal robot for sensorimotor tasks, *Tech. rep.*, University of Pennsylvania.
- Ghigliazza, R., R. Altendorfer, P. Holmes, and D. Koditschek (2004), A simply stabilized running model, *SIAM J. App. Dyn. Systems*, *2*(2), 187–218, doi:10.1137/S1111111102408311.

- Ghigliazza, R. M., and P. Holmes (2005), Towards a neuromechanical model for insect locomotion: Hybrid dynamical systems, *Regular & Chaotic Dynamics*, 10(2), 193–225, doi:10.1070/RD2005v010n02ABEH000311.
- Goebel, R., R. Sanfelice, and A. Teel (2009), Hybrid dynamical systems, *Control Systems, IEEE*, 29(2), 28–93, doi:10.1109/MCS.2008.931718.
- Golubitsky, M., and V. Guillemin (2012), *Stable mappings and their singularities*, vol. 14, Springer Science & Business Media.
- Groff, R. E., P. P. Khargonekar, and D. E. Koditschek (2003), A local convergence proof for the minvar algorithm for computing continuous piecewise linear approximations, *SIAM journal on numerical analysis*, 41(3), 983–1007.
- Guckenheimer, J. (), Isochrons and phaseless sets, *Journal of Mathematical Biology*, 1(3), 259–273, doi:10.1007/BF01273747.
- Guckenheimer, J., and S. Johnson (1995), *Hybrid systems II*, 202–225 pp., Springer-Verlag, London, UK.
- Gupta, K., and A. P. Pobil (1998), *Practical motion planning in robotics: Current approaches and future directions*, John Wiley & Sons, Inc.
- Hairer, E., G. Wanner, and P. Norsett (1993), Solving ordinary differential equations i – nonstiff problems, Springer Series in Computational Mathematics, 2 ed., Springer-Verlag.
- Hatton, R. L., and H. Choset (2011), An introduction to geometric mechanics and differential geometry.
- Hatton, R. L., and H. Choset (2013), Geometric swimming at low and high reynolds numbers, *IEEE Transactions on Robotics*, 29(3), 615–624.
- Hereid, A., E. A. Cousineau, C. M. Hubicki, and A. D. Ames (2016), 3d dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics, in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pp. 1447–1454, IEEE.
- Hill, A. (1938), The heat of shortening and the dynamic constants of muscle, *Proc. R. Soc. Lond. B*, 126(843), 136–195, doi:10.1098/rspb.1938.0050.
- Hirsch, M. W., R. L. Devaney, and S. Smale (1974), *Differential equations, dynamical systems, and linear algebra*, vol. 60, Academic press.
- Hirsch, M. W., C. C. Pugh, and M. Shub (2006), *Invariant manifolds*, vol. 583, Springer.
- Hirsch, M. W., S. Smale, and R. L. Devaney (2012), *Differential equations, dynamical systems, and an introduction to chaos*, Academic press.

- Holmes, P., R. Full, D. Koditschek, and J. Gukenheimer (2006a), The dynamics of legged locomotion : models, analyses, and challenges, *SIAM Review*, 48(2), 206–304, doi:10.1137/S0036144504445133.
- Holmes, P., R. Full, D. Koditschek, and J. Gukenheimer (2006b), The dynamics of legged locomotion : models, analyses, and challenges, *SIAM Review*, 48(2), 206–304, doi:10.1137/S0036144504445133.
- Hutter, M., R. C.D. M. Hoepflinger, and R. Seigwart (2010), Full state control of a slip model by touchdown detection, in *13th International Conference on Climbing and Walking Robots*, pp. 533–540.
- Ivanov, A. (1998), The stability of periodic solutions of discontinuous systems that intersect several surfaces of discontinuity, *Journal of Applied Mathematics and Mechanics*, 62(5), 677–685.
- Jarque-Bou, N., V. Gracia-Ibáñez, J.-L. Sancho-Bru, M. Vergara, A. Pérez-González, and F. Andrés (2016), Using kinematic reduction for studying grasping postures. an application to power and precision grasp of cylinders, *Applied ergonomics*, 56, 52–61.
- Jarque-Bou, N. J., A. Scano, M. Atzori, and H. Müller (2019), Kinematic synergies of hand grasps: a comprehensive study on a large publicly available dataset, *Journal of neuroengineering and rehabilitation*, 16(1), 63.
- Kelly, S. D., and R. M. Murray (1995), Geometric phases and robotic locomotion, *Journal of Robotic Systems*, 12(6), 417–431.
- Kenneally, G., A. De, and D. E. Koditschek (2016), Design principles for a family of direct-drive legged robots, *IEEE Robotics and Automation Letters*, 1(2), 900–907.
- Khalil, H. (2002a), *Non-Linear System: Third Edition*, Prentice Hall, Upper Saddle River, New Jersey.
- Khalil, H. (2002b), *Nonlinear systems*, 3 ed., Prentice Hall, Upper Saddle River, New Jersey.
- Kobayashi, S., and K. Nomizu (1963), *Foundations of differential geometry*, vol. 1, New York.
- Kober, J., J. A. Bagnell, and J. Peters (2013), Reinforcement learning in robotics: A survey, *The International Journal of Robotics Research*, 32(11), 1238–1274.
- Koditschek, D. E., and E. Rimon (1990), Robot navigation functions on manifolds with boundary, *Advances in applied mathematics*, 11(4), 412–442.
- Koon, W. S., and J. E. Marsden (1997), The geometric structure of nonholonomic mechanics, in *Proceedings of the 36th IEEE Conference on Decision and Control*, vol. 5, pp. 4856–4861, IEEE.



- Kuhn, H. W. (1960), Some combinatorial lemmas in topology, *IBM Journal of research and development*, 4(5), 518–524.
- Lee, J. M. (2013), Smooth manifolds, in *Introduction to Smooth Manifolds*, pp. 1–31, Springer.
- Levine, S., P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen (2018), Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection, *The International Journal of Robotics Research*, 37(4-5), 421–436.
- Lygeros, J., K. H. Johansson, S. N. Simic, J. Zhang, and S. S. Sastry (2003), Dynamical properties of hybrid automata, *IEEE Transactions on automatic control*, 48(1), 2–17.
- Marsden, J. E., R. Montgomery, and T. S. Ratiu (1990), *Reduction, symmetry, and phases in mechanics*, vol. 436, American Mathematical Soc.
- Meigniez, G. (2002), Submersions, fibrations and bundles, *Transactions of the American Mathematical Society*, 354(9), 3771–3787.
- Mordatch, I., Z. Popović, and E. Todorov (2012a), Contact-invariant optimization for hand manipulation, in *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*, pp. 137–144, Eurographics Association.
- Mordatch, I., E. Todorov, and Z. Popović (2012b), Discovery of complex behaviors through contact-invariant optimization, *ACM Transactions on Graphics (TOG)*, 31(4), 43.
- Müller, P. C. (1995), Calculation of lyapunov exponents for dynamic systems with discontinuities, *Chaos, Solitons & Fractals*, 5(9), 1671–1681.
- Murray, R., S. Sastry, and Z. Li (1994), *A mathematical introduction to robotic manipulation*, 1 ed., CRC Press.
- Murray, R. M. (2017), *A mathematical introduction to robotic manipulation*, CRC press.
- OLECH, C. (1998), On the wazewski equation, *ZESZYTY NAUKOWE-UNIwersytetu Jagiellońskiego-ALL SERIES-*, 1223, 55–64.
- Orhon, H. E. (2018), Model-based identification and control of a one-legged hopping robot, *arXiv preprint arXiv:1802.09634*.
- Ostrowski, J., and J. Burdick (1998), The geometric mechanics of undulatory robotic locomotion, *The international journal of robotics research*, 17(7), 683–701.
- Ostrowski, J. P. (1996), The mechanics and control of undulatory robotic locomotion, Ph.D. thesis, California Institute of Technology.

- Palmer III, L. R., and C. Eaton (2014), Periodic spring-mass running over uneven terrain using feedforward control of landing conditions, *Bioinspiration and Biomimetics*, 9(3), doi:10.1088/1748-3182/9/036018.
- Pepy, R., A. Lambert, and H. Mounier (2006), Path planning using a dynamic vehicle model, in *2006 2nd International Conference on Information & Communication Technologies*, vol. 1, pp. 781–786, IEEE.
- Pepyne, D. L., and C. G. Cassandras (2000), Optimal control of hybrid systems in manufacturing, *Proceedings of the IEEE*, 88(7), 1108–1123.
- Pratihari, D. K., K. Deb, and A. Ghosh (2002), Optimal path and gait generations simultaneously of a six-legged robot using a ga-fuzzy approach, *Robotics and Autonomous Systems*, 41(1), 1–20.
- Radford, J., and J. Burdick (1998), Local motion planning for nonholonomic control systems evolving on principal bundles, *A, A, 1, 3*.
- Ramezani, A., J. W. Hurst, K. A. Hamed, and J. W. Grizzle (2014), Performance analysis and feedback control of atrias, a three-dimensional bipedal robot, *Journal of Dynamic Systems, Measurement, and Control*, 136(2), 021,012.
- Ratliff, N., M. Zucker, J. A. Bagnell, and S. Srinivasa (2009), Chomp: Gradient optimization techniques for efficient motion planning, in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pp. 489–494, IEEE.
- Reinkensmeyer, D. J., et al. (2014), Tools for understanding and optimizing robotic gait training.
- Reist, P., and R. D’Andrea (2012), Design and analysis of a blind juggling robot, *IEEE Transactions on Robotics*, 28(6), 1228–1243.
- Revzen, S., and J. Guckenheimer (2008), Estimating phase of synchronized oscillators, *Phys. Rev. E*, 78(5), 051,907, doi:10.1103/PhysRevE.78.051907.
- Revzen, S., and M. Kvalheim (2015), Data driven models of legged locomotion, in *SPIE Defense+ Security*, pp. 94,671V–94,671V, International Society for Optics and Photonics.
- Revzen, S., D. E. Koditschek, and R. J. Full (2009), Towards testable neuromechanical control architectures for running, in *Progress in Motor Control*, pp. 25–55, Springer.
- Revzen, S., M. Kvalheim, S. Wilshon, and J. Guckenheimer (2018), Estimating phase from observed trajectories, *In preparation*, pp. 1–20.
- Rockafellar, R. T. (2003), A property of piecewise smooth functions, *Computational Optimization and Applications*, 25(1-3), 247–250.
- Ruff, C. (2002), Variation in human body size and shape, *Annual Review of Anthropology*, 31, 211–232, doi:10.1146/annurev.anthro.31.040402.085407.

- Rumbaugh, J., I. Jacobson, and G. Booch (2004), *Unified modeling language reference manual, the*, Pearson Higher Education.
- Saranli, U., M. Buehler, and D. E. Koditschek (2001), Rhex: A simple and highly mobile hexapod robot, *The International Journal of Robotics Research*, 20(7), 616–631.
- Sastry, S. (1999), *Nonlinear systems : analysis, stability, and control*, Springer.
- Schmitt, J., and P. Holmes (2000), Mechanical models for insect locomotion: dynamics and stability in the horizontal plane i. theory, *Biological cybernetics*, 83(6), 501–515.
- Scholtes, S. (2012), *Introduction to piecewise differentiable equations*, Springer Science & Business Media.
- Seyfarth, A., H. Geyer, M. Gunther, and R. Blickhan (2002), A movement criterion for running, *J. Biomechanics*, 35, 649–655, doi:10.1016/S0021-9290(01)00245-7.
- Seyfarth, A., H. Geyer, and H. Herr (2006), Swing-leg retraction : a simple control model for stable running, *J. of Exp. Bio.*, 206, 2547–2555, doi:10.1242/jeb.00463.
- Sharir, M. (1989), Algorithmic motion planning in robotics, *Computer*, 22(3), 9–19.
- Siepel, J., and P. Holmes (2007), A simple model for clock-actuated legged locomotion, *Regular and Chaotic Dynamics*, 12(5), 502–520, doi:10.1134/S1560354707050048.
- Simić, S. N., K. H. Johansson, S. Sastry, and J. Lygeros (2000), Towards a geometric theory of hybrid systems, in *International Workshop on Hybrid Systems: Computation and Control*, pp. 421–436, Springer.
- Spivak, M. (1965), *Calculus on Manifolds*, Perseus Books Publishing, L. L. C.
- Spröwitz, A., A. Tuleu, M. Vespignani, M. Ajallooeian, E. Badri, and A. J. Ijspeert (2013), Towards dynamic trot gait locomotion: Design, control, and experiments with cheetah-cub, a compliant quadruped robot, *The International Journal of Robotics Research*, 32(8), 932–950.
- Tassa, Y., T. Erez, and E. Todorov (2012), Synthesis and stabilization of complex behaviors through online trajectory optimization, in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4906–4913, IEEE.
- Tedrake, R., T. W. Zhang, and H. S. Seung (2004), Stochastic policy gradient reinforcement learning on a simple 3d biped, in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3, pp. 2849–2854, IEEE.
- Utkin, V. (1977), Variable structure systems with sliding modes, *IEEE Transactions on Automatic control*, 22(2), 212–222.

- Vejdani, H., Y. Blum, M. Daley, and J. Hurst (2013), Bio-inspired swing leg control for spring-mass robots running on ground with unexpected height disturbance, *Bioinspiration and Biomimetics*, 8(4), doi:10.1088/1748-3182/8/4/046006.
- Westervelt, E. R., J. W. Grizzle, and D. E. Koditschek (2003), Hybrid zero dynamics of planar biped walkers, *IEEE transactions on automatic control*, 48(1), 42–56.
- Winters, J. (1990), Hill-based muscle models: a systems engineering perspective, in *Multiple Muscle Systems*, edited by J. Winters and S.-Y. Woo, pp. 69–93, Springer New York, doi:10.1007/978-1-4613-9030-5\_5.
- Yosinski, J., J. Clune, D. Hidalgo, S. Nguyen, J. C. Zagal, and H. Lipson (2011), Evolving robot gaits in hardware: the hyperneat generative encoding vs. parameter optimization., in *ECAL*, pp. 890–897.
- Yu, H., M. Li, W. Guo, and H. Cai (2012), Stance control of the slip hopper with adjustable stiffness of leg spring, in *Mechatronics and Automation (ICMA), 2012 International Conference on*, pp. 2007–2012, IEEE.

## CHAPTER IV

# Deadbeat Stabilization

### 4.1 Deadbeat Hopping

In limbed biological and robotic systems, intermittent contacts that arise frequently are defining feature. Feet make and break contact with the ground while walking or running, while grasping motions are practically defined by such events. However, mechanical contacts with the ambient environment are subject to uncertainty in the contact timing. Perfect knowledge the environment is not a luxury practical systems have. The environment is uneven, terrain is rugged, objects are not uniformly shaped, and so on. Similarly, no model is ideal – body configuration and morphology are not known exactly. A robot is not manufactured to infinite precision, and organisms exhibit variation in body parameters across members of a species and even through its lifetime ([Ruff, 2002](#)). As such, contacts with the environment during manipulation or locomotion occurs at an unexpected time, which appear as exogenous disturbances from the perspective of the actor (Fig. 4.1). To enable robust activity, such as reliable locomotion over rough ground or sure gripping of variably-shaped objects, limbs and manipulators need a method to compensate for these uncertain contacts.

We would like to design a principled method to mitigate the effects of unexpected contacts. We immediate specialize to rejecting uncertain contacts that occur in hybrid

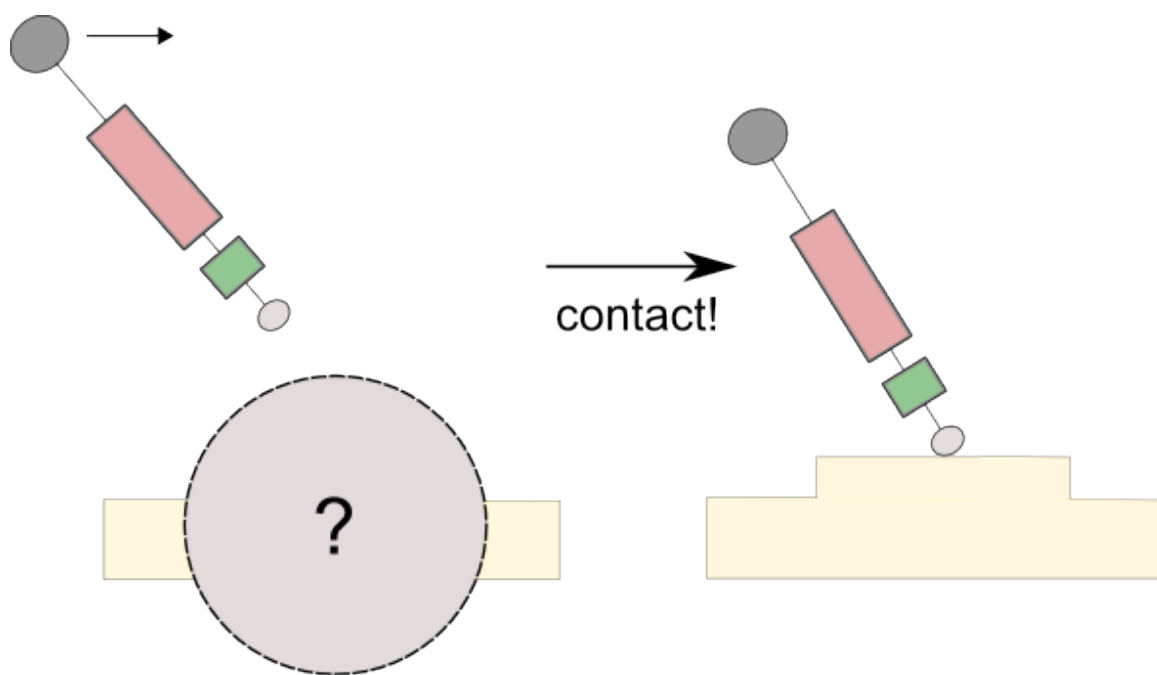


Figure 4.1: A depiction of a SLIP model in gravity subject to uncertain contact. While a window of possible contacts is known, the precise point of contact is not. The control method proposed here presents a solution to preserve the gait of the SLIP.

dynamical systems, which is a commonly used class of models for systems that experience discontinuities as a result of contact ([Full and Koditschek, 1999](#); [Goebel et al., 2009](#)). The applicability of hybrid models to contacting systems is well-established ([Holmes et al., 2006b](#); [Goebel et al., 2009](#)). The governing dynamics prior to contact and post-contact are assumed to be different smooth models, where the contact is a simplifying abstraction that accommodates fast time-scale dynamics at the transition surface.

Our high-level concern is how to achieve a post-contact state reliably despite uncertainty in when a transition manifold is crossed. While the control method is made precise in §4.1.4, the essential perspective is *trajectory selection*, rather than *sensing*.

We wish to achieve our goal without control effort post-contact. Typically, limbs and manipulators have a payload capacity. As a consequence, they are often easy to control before contact, but become load-limited when in contact. This change expresses itself in a dramatic reduction in control authority; even if the system remains fully controllable in a formal sense, the amount of actuator effort rises significantly.

Since the objective is to mitigate contacts with the environment, which is external to the actor, it is only natural to conjecture that a substantial amount of sensing and control is needed if disturbances from exogenous contact changes are to be rejected. The essence of our contribution is showing this conjecture to be false both in an example, and almost always under mild assumptions. We demonstrate a means whereby a desired control law for the outcome in the poorly controllable and load-limited region can be encoded in a manifold in the region where the uncertain transitions may occur. If the pre-transition control authority can be used to render the manifold invariant, the actor can remove most or all the uncertainty generated by an exogenously driven hybrid transition. Roughly, we do this by computing a solution at each possible transition event whose outcome achieves the desired end-state, defining a collection

of state/transition pairs that map to a desired outcome. If the set of pairs can be parameterized as a function of the pre-contact state, control action in the pre-contact region can implement the discovered function, rendering the outcome of the system deadbeat invariant with respect to the transition (Fig. 4.2).

The notion that feedforward controllers might improve the behavior of SLIP models was partially motivated by [Seyfarth et al. \(2006\)](#), where Seyfarth et. al. demonstrated that a blind retraction of a bipedal leg during descent improved the stability properties of a periodic gait, but did not present a prescriptive technique for general systems. While the numerical examples we consider are both SLIP-variants, the geometric method introduced here is a generic result for hybrid systems. We acknowledge that a significant number of other groups ([Ernst et al., 2011](#); [Palmer III and Eaton, 2014](#); [Hutter et al., 2010](#); [Vejdani et al., 2013](#); [Orhon, 2018](#)) have presented feedforward controllers for SLIP models to stabilize a desired periodic motion with respect to uncertain ground contacts in either simulation or with hardware implementations. Similar strategies have been used for manipulation tasks, such as juggling ([Reist and D’Andrea, 2012](#)).

Our proposed contribution two-fold. First, we present a precise mathematical statement that is a generalized representation of the deadbeat control problem, and argue that, generically, many such hybrid systems could admit such structure, whereas the cited work ostensibly made use of SLIP-specific model structure. Secondly, as we are fundamentally interested in an finite-duration trajectories (periodic orbits for a single period fit into this category), we argue, and then demonstrate via simulation, that deadbeat feedforward controllers can be *derived from trajectory data* without the need for equations of motion. The aforementioned authors invariably used a dynamic model either derived from first principles, or fit to experimental data.

Our results imply that, beyond generating a controller that regulates a plant in an actor-plant model, the independence of the feedforward controller allows direct



incorporation into the plant. Such a “co-design” of plant and controller is a step towards embodied intelligence, where favorable, task-specific, dynamics can be directly incorporated into the mechanical structure of a limb. The observation suggests a methodology to design robots, as well as provide a new perspective on the analysis of animal morphology, where neural processing and mechanical structure are not independently designed. Biologists and biomechanists who seek to identify the mechanisms of control in animals should be wary of the possibility that critically important control mechanisms may be encoded in the geometry of limbs and the choice of feed-forward limb trajectory, partially or completely bypassing the nervous system.

#### 4.1.1 Periodic Hopping

We are attempting to determine whether we can control the outcomes of such a model with little to no sensing, by choosing appropriate feed-forward actions while a leg is still in the air. Representing these motions in a body-centric frame of reference, foot touchdown can typically occur anywhere within the legs’ workspace. We therefore consider hybrid systems with an open “guard” – a region in which an exogenous transition may occur – rather than the more commonly used formulation of guards as submanifolds with boundary in the boundary of a domain ([Goebel et al., 2009](#)).

Choosing, as many legged locomotion papers do ([Seyfarth et al., 2002, 2006](#); [Arslan and Saranli, 2012](#); [Carver et al., 2009](#); [Ankarali and Saranli, 2010](#)), to analyze the periodic motion of our hopping model using Poincaré return maps to the apex of the hop, we will show a means to select foot motion in the neighborhood of an expected touchdown location so that the model performs deadbeat perturbation rejection for ground height changes. This means that our model will perfectly track changes in ground height by adjusting the subsequent apex height by the same amount, but it will do so *without sensing the ground*. Inside, the method relies entirely on the interaction of the touchdown event with our choice of foot trajectory to produce the

“control”. The only sensing our model uses is knowledge of how far it has fallen since starting apex. In this sense, we only need to know the initial state, and measure the elapsed time, which we consider to be proprioceptive sensing.

#### 4.1.2 Encoding control as a pre-transition constraint manifold

Numerous definitions of hybrid dynamical systems appear in the literature ([Sastry, 1999](#); [Guckenheimer and Johnson, 1995](#); [Goebel et al., 2009](#)). Since we seek to show a local result, we will focus on the flow across a single “hybrid transition”. By stating our result with respect to a single, possibly discontinuous, change in the equations of motion governing the dynamics we avoid the need for mathematical machinery defining a specific, possibly overly restrictive, class of hybrid systems.

#### 4.1.3 Hybrid control system definition

Consider a system of the form:

$$\dot{x} = \begin{cases} f_{\mathcal{A}}(x, u), & x \in \mathcal{A}, f_{\mathcal{A}} : \mathcal{A} \times \mathbb{R}^k \rightarrow \mathbf{T}\mathcal{A} \\ f_{\mathcal{B}}(x), & x \in \mathcal{B}, f_{\mathcal{B}} : \mathcal{B} \rightarrow \mathbf{T}\mathcal{B} \end{cases} \quad (4.1)$$

with vector fields  $f_{\mathcal{A}}, f_{\mathcal{B}}$  Lipschitz in  $x$  on the closures of their respective precompact domains  $\mathcal{A}, \mathcal{B}$ , which map into the respective tangent bundle, and a control input  $u$  entering into  $f_{\mathcal{A}}$  from  $\mathbb{R}^k$ . Assume it is furthermore known that all executions<sup>1</sup> starting in an open set  $U_0 \subseteq \mathcal{A}$  will flow into an open set  $O \subseteq \mathcal{A} \cap \mathcal{B}$  and somewhere therein switch to the  $\mathcal{B}$  domain, to continue with the  $\mathcal{B}$  dynamics into an open set  $U_1 \subseteq \mathcal{B}$ . We will presume that the switching between  $\mathcal{A}$  and  $\mathcal{B}$  dynamics may be exogenous: it might not be a function of state, nor of time, nor be in any other way amenable to prediction; we are only given that exactly one such switching event will occur for every execution starting at  $U_0$ , and that this transition will occur while the

---

<sup>1</sup>a “solution” of a hybrid system is termed an “execution”

state is in  $O$ .

#### 4.1.4 Controller design

Assume that for the system of Eqn. (4.1.3) we are furnished with transverse Poincaré section  $S \subseteq U_1$  given as the 0 level set of a smooth function  $\sigma : U_1 \rightarrow \mathbb{R}$ , i.e.  $S := \sigma^{-1}(0)$ , and  $D_x \sigma \cdot f_{\mathcal{B}} > 0$  in the entire domain it is defined. The section  $S$  will be used to define the output we wish to control, via a smooth function  $g : O \times S \rightarrow \mathbb{R}^d$  whose Jacobians  $D_S g$  and  $D_O g$  with respect to both  $S$  and  $O$  are maximal rank everywhere on  $S$  and  $O$  respectively. We will use  $g$  to define an implicit relationship between transition states  $O$  and the desired eventual states in  $S$ , such that for each execution  $y$  with transition at  $y_\tau$  and arrival at  $y_1 \in S$ , we drive the value of  $g(y_\tau, y_1)$  to zero, like a cost function from optimization.

As one final assumption, we must develop our local control scheme in the neighborhood of a known “desirable” execution of the system  $x : [0, 1] \rightarrow (\mathcal{A} \cup \mathcal{B})$  for which the control  $u$  is identically 0.  $x$  starts at  $x(0) = x_0 \in \mathcal{A}$ , moves into the open set  $O \subseteq \mathcal{A} \cap \mathcal{B}$  in which it transitions over at time  $\tau_*$  and state  $x_* := x(\tau_*)$  to  $\mathcal{B}$ , then continues in  $\mathcal{B}$ , finishing at  $x(1) = x_1 \in S$ . It is “desirable” in the sense that  $g(x_*, x_1) = 0$ .

Although we have no control over the dynamics in  $B$ , nor over the instant at which  $f_{\mathcal{B}}$  takes over the dynamics, we will show that we can *locally deadbeat stabilize* the output  $g = 0$  by restricting the trajectories in  $O$  to a smooth submanifold passing through  $x_*$ , using the control  $u$ . This follows from the observation that the flow  $\Phi_{\mathcal{B}} : \mathbb{R}^+ \times \mathcal{B} \rightarrow \mathcal{B}$  in domain  $\mathcal{B}$  is smooth in  $\mathcal{B}$ , as is the “impact time map”  $T_{\mathcal{B}} : O \rightarrow \mathbb{R}^+$  ([Hirsch et al., 1974](#); [Arnold, 1973](#)) which maps initial conditions to their arrival times on  $S$ , i.e.  $\forall o \in O : \Phi_{\mathcal{B}}(T_{\mathcal{B}}(o), o) \in S$ , which is easily defined with

the Implicit Function Theorem. We pull  $S$  and thus also  $g$  back to  $O$  by defining:

$$g^*(x) := g(x, \Phi_{\mathcal{B}}(T_{\mathcal{B}}(x), x)) \quad g^* : O \rightarrow \mathbb{R}^d. \quad (4.2)$$

By virtue of being the pullback,  $g^*$  maps every state in  $O$  to its “eventual” output value, presuming that it will be carried by  $\Phi_{\mathcal{B}}$  for the remaining duration.

Consider a path  $z(t)$  taking values in  $O$ , and which is at least  $\mathcal{C}^1$ . The derivative of  $g^*$  along the path  $z$  is, via the Chain Rule:

$$\frac{d}{dt}g^*(z(t)) = [\mathbf{D}_O g + \mathbf{D}_S g \cdot (\mathbf{D}_x \Phi_{\mathcal{B}} + f_{\mathcal{B}} \cdot \mathbf{D}_x T_{\mathcal{B}})] \cdot \dot{z}. \quad (4.3)$$

If Eqn. (4.3) equals zero for all time, then the path  $z(t)$  is  $g_*$ -invariant. We will assume  $g^*$  is of rank  $d$  in a neighborhood  $X_*$  of  $x_*$  and conclude that the set

$$\mathcal{G} := \{x \in X_* \mid g^*(x) = 0\} \quad (4.4)$$

is an embedded submanifold of co-dimension  $d$  in  $O$ , which passes through  $x_*$ , since  $g^*(x_*) = 0$ .

If a controller can bring trajectories  $z$  in  $\mathcal{A}$  close to  $\mathcal{G}$ , i.e. to  $\|g^*(z)\| \leq \varepsilon$ , and thereafter sustain the  $\frac{d}{dt}g^*(z(t)) = 0$ , then any transition that occurs in  $X_*$  will lead to a state with  $\|g\| \leq \varepsilon$ . In particular, this is true even for  $\varepsilon = 0$ , in which finite time convergence to  $\mathcal{G}$  will ensure that  $g = 0$  after transition.

A range of possible methods of nonlinear control ([Khalil, 2002b](#); [Sastry, 1999](#)) may be employed to achieve this goal in  $\mathcal{A}$ . The choice of controller is not germane to the core observation we present. Namely, that  $\mathcal{G}$  is known in advance, independently of any need for sensing in real time. In its geometry it encodes a relationship between the transition state in  $O$  and the outcome on the next crossing of  $S$  in a feed-forward fashion.

As a corollary, note that if  $g$  is of maximal dimension, i.e. if  $d = \dim \mathcal{B} - 1 = \dim S$ , then  $\mathcal{G}$  is a single smooth trajectory passing through  $x_*$ . When moving along this trajectory, the system is guaranteed to reach  $x_1$  in  $S$  regardless of where in  $O$  the transition occurs. If, furthermore, the flow carries  $\mathcal{B}$  back into  $\mathcal{A}$  such that the example execution is periodic (stable or not), this process renders the controlled execution  $z$  stable with respect to the Poincaré section defined by  $S$ . Any deviations from the nominal  $z$  will be removed by the controller acting in  $\mathcal{A} \setminus O$ . We now proceed to construct such a deadbeat control example in simulation.

#### 4.1.4.1 Vertical hopping can be Deadbeat controlled without sensing

As a motivating example, we present a model of a vertically hopping organism. In the classical paper [Blickhan \(1989a\)](#), Blickhan defined a vertical spring-mass hopper and proceeded to derive relationships for the performance envelopes of human hoppers and runners. In line with Blickhan, Farley ([Farley et al., 1985](#)) has argued that the vertical hopping limit of the Spring Loaded Inverted Pendulum (SLIP) model of running ([Holmes et al., 2006a](#); [Blickhan, 1989a](#); [Ghigliazza et al., 2004](#)) – running with zero forward speed – is informative in understanding bouncing gaits in animals. We suggest some simple elaborations of this model can provide a platform for demonstrating our key result, while at the same time providing a more general model capturing some of the essential nature of control of legged systems with muscles.

A fundamental property of both the classic SLIP and Blickhan’s hopper is that they are energy conserving systems. As such, even if a periodic execution exists, it cannot be asymptotically stable – any perturbation changing total energy must perforce shift the system to a new level-set of the total energy function, never to return. Changes in average ground height, which are of paramount relevance to any legged locomotion in the real world, will require changes in average potential energy. We thus require a dynamical model which allows for energy to be both added and

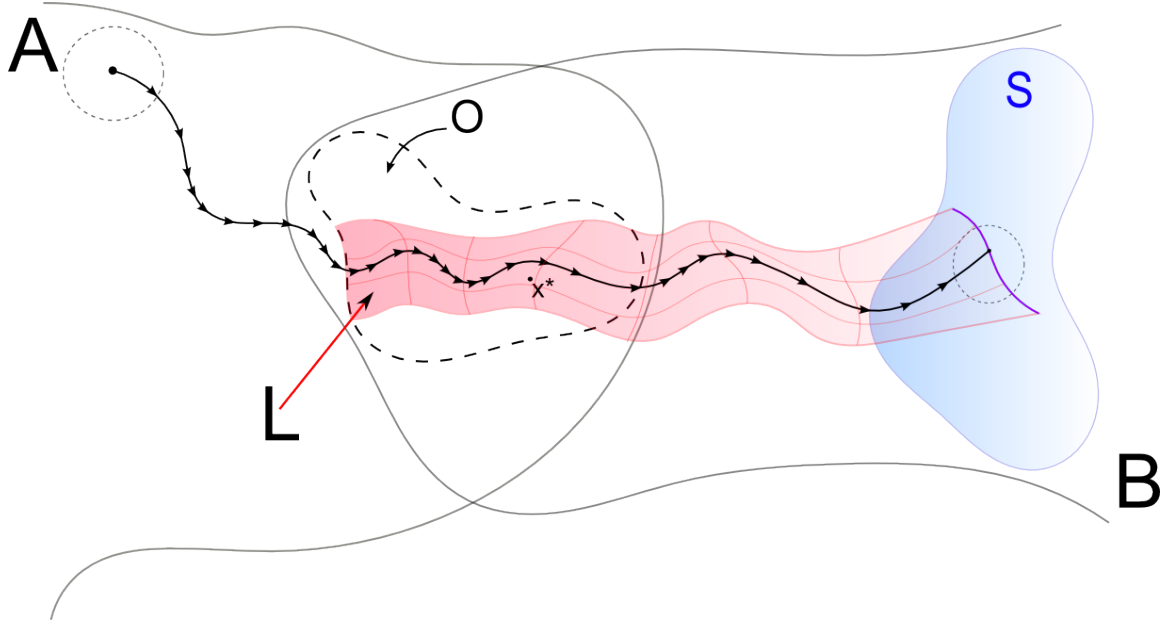


Figure 4.2: A visual depiction of arrangement of hybrid domains. The submanifold  $L$  (the zero level set of the goal surface, pulled back via the flow in  $B$ ) is defined by the flow of the vector field defined in domain  $B$ . If the dynamics of the  $A$  system naturally have dynamics invariant to this surface, it will reject any uncertainty in switching manifold.

removed from the system.

Since our goal is to provide an example with no sensing of the ground, the approach of energy injection by triggering some actuation whilst on the ground cannot serve us. Instead, we hark back to biology to note that biological actuation uses muscles, whose intrinsic dynamics already allow for them to add energy and not only remove it. Numerical experimentation with our model has shown this to require some “toe-off” – the liftoff height in hopping must be greater than the landing height. This result of our model may deserve further biomechanical investigation; it seems plausible for human hopping that ground reaction forces only become substantial at heel strike, but persist to toe-off.

The overall scheme of our model is illustrated in Fig. 4.3, showing a hopper hopping over changing ground. We envision the hopper to have an actuator allowing the length of the leg to be freely controlled in flight, but presume this leg length to

“lock in” at touchdown, and have dynamics of stance governed entirely by a muscle model, until the length of the leg exceeds the toe-off length, or until the relative height is 0 (the leg “crashed”).

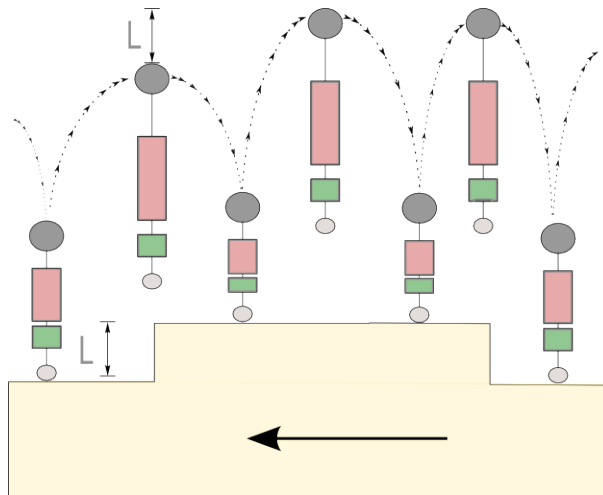


Figure 4.3: Vertical Hopper bouncing over a moving plate (height plotted against time), with an actuator (active only in flight) changing the rest length of the leg (pale red) and a muscle-like element (green) supporting the payload mass (dark grey). While the foot (light grey) is in contact with the ground, ground height remains constant; it changes instantaneously when the hopper is at its apex height.

#### 4.1.4.2 Equations of motion

The state of our hopper is defined by the (absolute) vertical position  $x$  and velocity  $\dot{x}$ . Its leg length, in flight, is governed directly by the control input  $u$ . Ground height  $h$  is piecewise constant, changing only when the hopper is at “apex” – in the flight domain, at the instant when  $\dot{x} = 0$ . We use the apex as our section  $S$ , with  $-\dot{x}$  playing the role of  $\sigma$  from the definition of  $S$ .

In flight, dynamics are ballistic:

$$\ddot{x} = -g \tag{4.5}$$

In stance, dynamics are governed by a combination of gravity and “muscle dy-

namics”.

$$\ddot{x} = -g + K(L - x)(1 - \eta\dot{x}) - \mu\dot{x} \quad (4.6)$$

The length  $L$  is the length of the leg at touchdown. The parameters  $K$  and  $\eta$  provide averaged approximations to the length dependent and velocity dependent terms (respectively) of the Hill muscle model ([Hill, 1938](#));  $\mu$  adds some dissipation, capturing the overall energy consuming nature of the task. The Hill muscle model has been postulated to be of sufficient accuracy to be useful for simulating human musculoskeletal behavior [Winters \(1990\)](#); [Bogert et al. \(1998\)](#). We base our treatment of this model on the presentation in §2 of [Ghigliazza and Holmes \(2005\)](#). With respect to their notation,  $K$  is an average slope for  $F_L$  near the operating point of the hopper, and  $\eta$  is an average slope for  $F_V$  around 0, and within the typical range of velocities.

The length of the leg is assumed to be directly controlled in the air, and is indicated by  $u \in \mathbb{R}$ . Then, touchdown transitions triggering a change from flight to stance are induced when:

$$x_{TD} - u = h \quad (4.7)$$

At that moment, the value of  $L$  is set to equal  $x_{TD} - h$  for the following stance.

Liftoff transitions are triggered when toeff occurs, i.e.,

$$x_{LO} = L + l_{to} \quad (4.8)$$

A “toeff” length  $l_{to} := 0.1$  was used for all our simulations.



#### 4.1.4.3 Simulation environment

The system was simulated in Python 2.7.5 using the NumPy and SciPy open-source numerical libraries. ODE-s were integrated using a pure-python port of the `dopri5` integrator from [Hairer et al. \(1993\)](#). A custom integrator was used because this integrator provides a “dense output”, i.e. polynomial patches between timepoints. These were used to implement a bisection based event detector, allowing for accurate and speedy simulation of hybrid systems.

#### 4.1.4.4 Constructing the goal function $g^*$

As a first step towards constructing an example of our control scheme, we produced a periodic solution of the hopping simulation by numerically solving for a fixed point of the apex return map.

Parameters of this solution are found in Table 4.2.

Table 4.1: Parameters

Parameter	Definition	Value
$\eta$	$F_V$ average slope	0.03
$\mu$	dissipative loss	0.3
$K$	$F_L$ average slope	80
$y_1$	desired relative apex	2
$y_*(y, \dot{y})$	known touchdown	(1.57, -2.87)
$l_{to}$	toe-off height	0.1

It is convenient to visualize the hopper in terms of its motions relative to the ground, rather than its dynamics with respect to absolute ground height. For this we define auxiliary coordinates  $y := x - h$ , whose dynamics are identical to those of  $x$  except for being discontinuously remapped by the change in ground height at apex.

Our goal for control is to ensure that hopping dynamics remain the same with respect to ground height, which for this very simple system implies that we require  $y$

at the following apex to be unchanged even if the landing height  $L$  was unexpected. This is expressed by the goal function:

$$g(y_{TD}, \dot{y}_{TD}, y_{next}) = (y_{TD} - y_*) - (y_{next} - y_1) \quad (4.9)$$

where  $y_{TD}$ ,  $\dot{y}_{TD}$ ,  $y_{next}$  are the touchdown state and the next apex height, and  $y_*$  and  $y_1$  are the touchdown and apex heights on the desired periodic execution. The choice of Eqn. (4.9) as  $g$  implies that touchdown height changes exactly with the subsequent apex height.

Using Eqn. (4.9) and the flow map, we sampled  $g^*$ , to obtain its values in a neighborhood of the expected touchdown state  $(\dot{y}_*, y_*)$  (see Fig. 4.4).

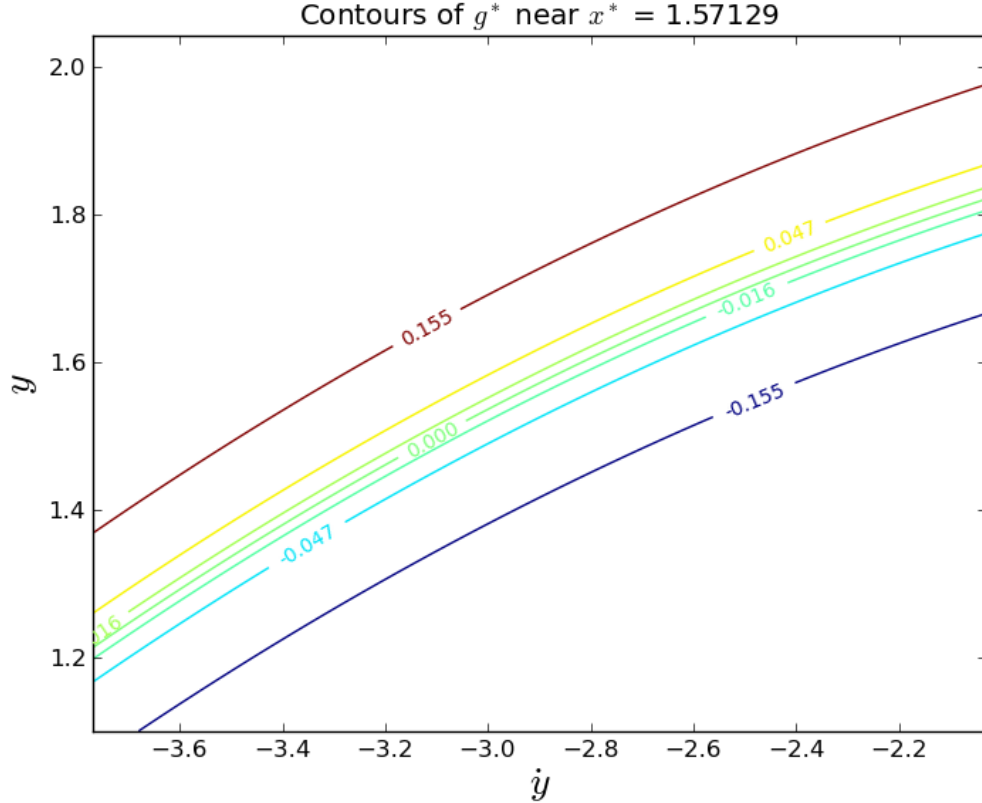


Figure 4.4: The pullback of  $g$  into  $g^*$  into the region  $X^*$  around  $y_* = 1.571$ . The zero contour is the control objective in this region. Values were sampled on a grid with step size 0.012

#### 4.1.4.5 Constructing the feed-forward control law

To numerically identify  $\mathcal{G}$ , the zero level set of  $g^*$ , we selected only points whose  $g^*$  value was less than 0.01 in magnitude, and least-squares fit a quadratic to obtain  $y$  as a function of  $\dot{y}$  and thereby indirectly as a function of time since last apex, or distance below last apex. To extend this function outside the sampled region, we held the extremal values using an analytical approximation to a step function. The final control law, written in terms of time since last apex  $t$ , the polynomial model `uModel`, the minimal and maximal sampled  $y$  values `y_min` and `y_max` and the minimal and maximal  $\dot{y}$  as `v_min`, `v_max` was:

---

```
def aStep( x ):
    return (tanh(x)+1)/2

def u(t):
    v = -g*t
    x0 = y1 - g*t*t/2
    w0 = aStep((v_min-v)/0.01)
    w1 = aStep((v-v_max)/0.01)
    w = (1-w0-w1)
    return (polyval(uModel,v)*w+y_min*w0+y_max*w1)+x0
```

---

Figs. 4.5 and 4.6 show typical simulation results.

The non-conservative nature of the control is evident in the simulation results depicted in Figs. 4.5 and 4.6. The hopper is gaining and losing energy from hop to hop in the precise amount needed to maintain a fixed *relative* displacement with respect to the ground at apex.

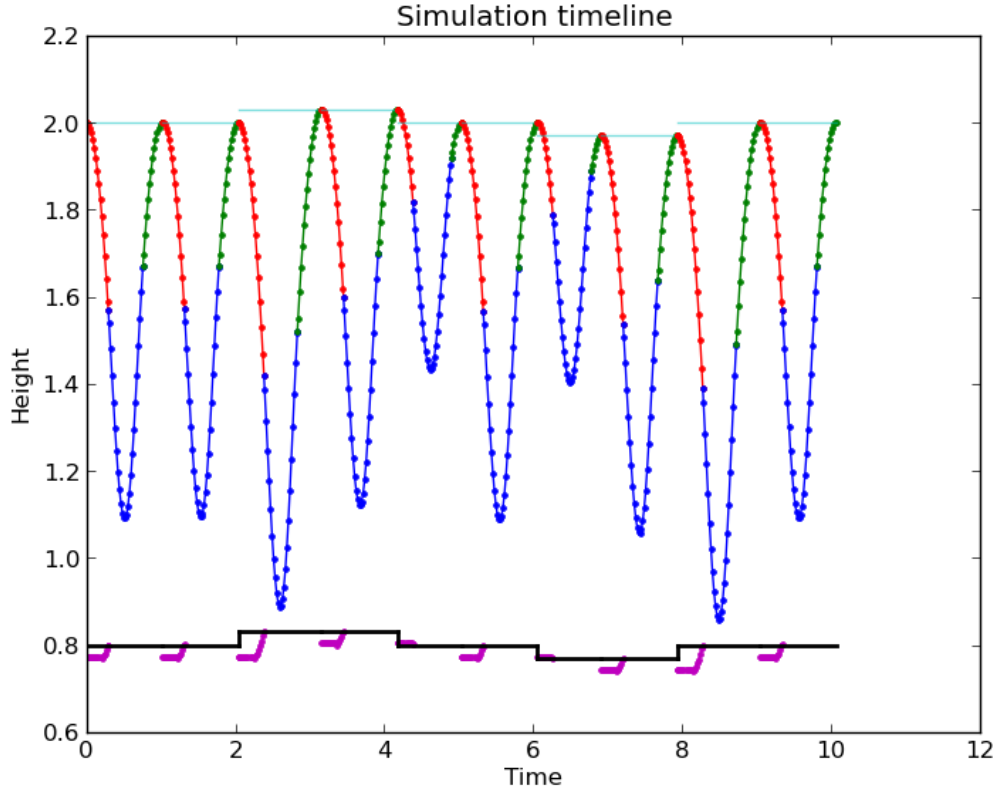


Figure 4.5: Hopper simulation with our controller and randomly changing ground height. Each hop begins with a descent (red) associated with a changing “virtual” foot position (purple), ending when the foot intercepts the ground, triggering a transition into stance (blue), which persists until toe-off. After toe-off the hopper ascends (green) to a new apex; upon reaching it the ground height (black) may change. To demonstrate the deadbeat nature of this controller, the desired apex height for the next hop is indicated (light blue), and each ground height persists for two hops to show that hopping height reaches a new steady state after a single hop.

#### 4.1.4.6 Physical realization of the model

Fig. 4.7 illustrates both the control objective graphically, and a potential difficulty with physical realization. The state is *below* the  $g_*$  contour, which therefore requires the leg to be shortened. Viewed naively, such a controller requires the toe to start “inside” the ground, and have the leg continually shortening until touchdown. Since the leg length we use is the length of the leg at the onset of generation of ground reaction forces,  $x_*$ , one may equally consider this shortening leg as a change of internal

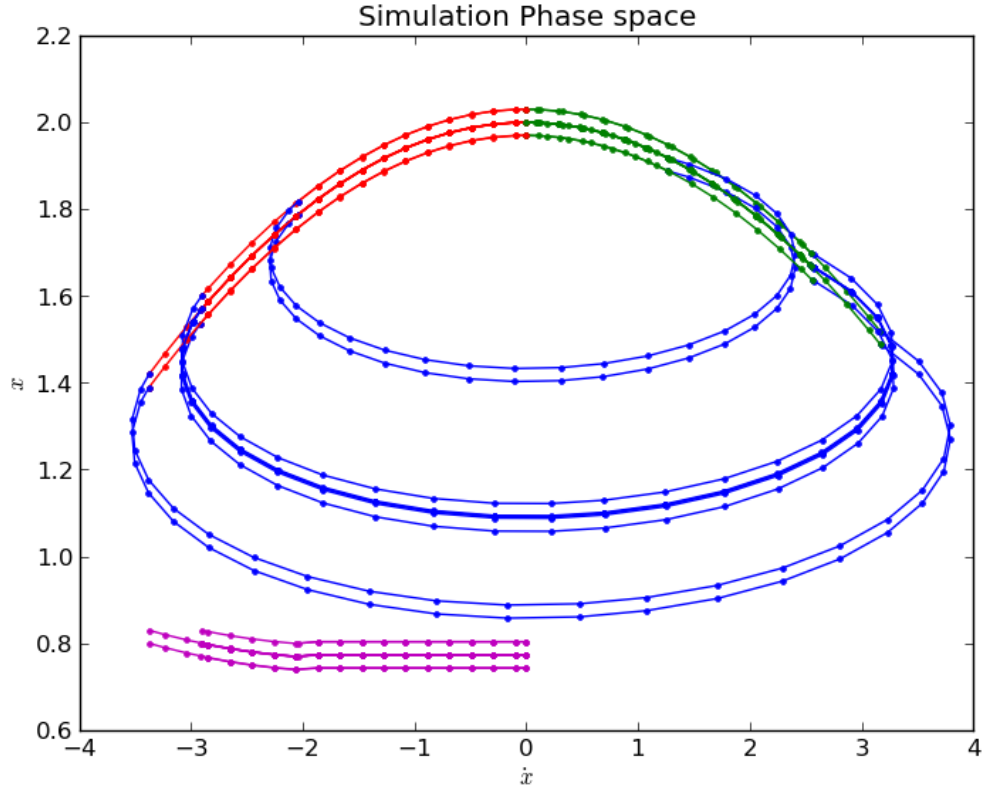


Figure 4.6: Phase-space plot of the same simulation as Fig. 4.5

configuration in a mechanism longer than  $L$ , which causes it to start producing force at the desired location.

#### 4.1.4.7 Implications

The above demonstration of the local deadbeat stabilization of a hopping model illustrates that accurate sensing of the ground is not a prerequisite for producing strong stability guarantees with respect to ground height changes. Given the existence of a periodic solution, a controller that only has limited sensing information – resets of a time-course at apex events – is able to provide a deadbeat stabilizing region of attraction and reject variations in ground height.

Importantly, given the “open-loop” nature of our control scheme one might consider that mechanical implementation of trajectories tangent to  $\mathcal{G}$  would provide

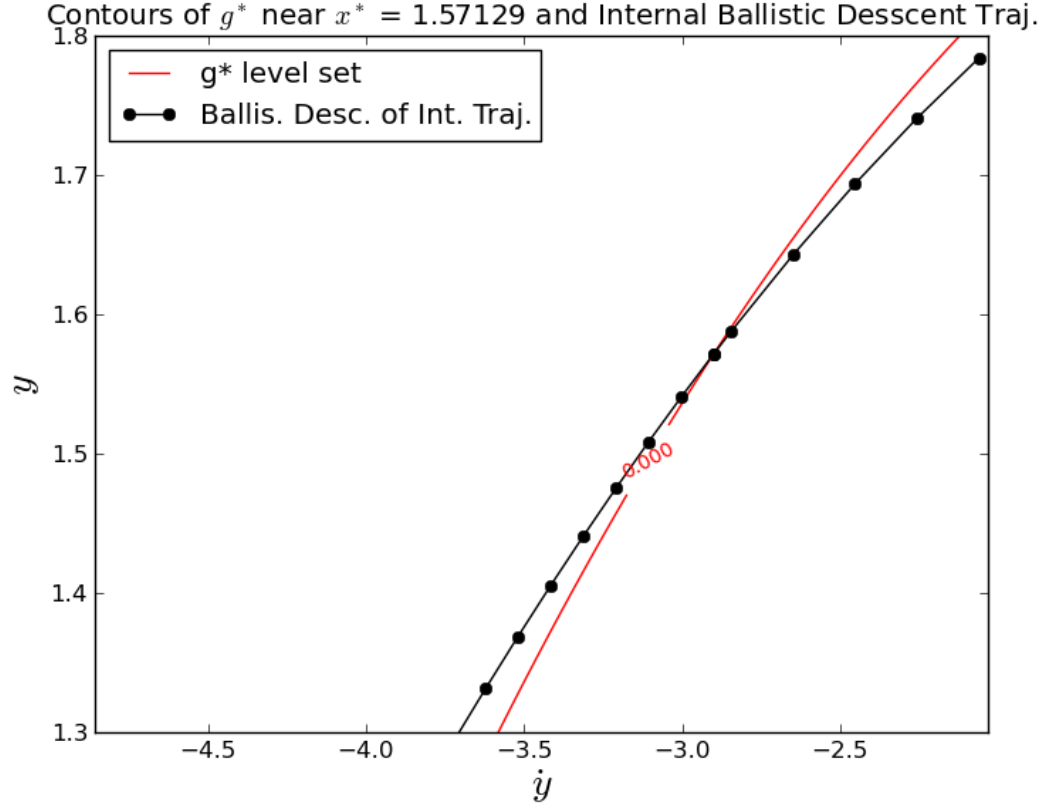


Figure 4.7: The relative position of the internal state and the pre-image of 0 under  $g_*$ . The difference between the two illustrates the necessary leg length to shift the descent trajectory pointwise to the contour.

strong self-stabilization without need for any overt sensing. Because the stabilization mechanism is encoded indirectly in the dynamics of the system, it would not be obvious even to a relatively sophisticated observer of the animal or robot system. While in our example the control objective was achieved entirely via trajectory design, a low value of  $d$  would merely act to produce some strongly stable (e.g., deadbeat) directions – allowing partial encoding of the control objective.

While maintaining  $\dot{g} = 0$  in a neighborhood of  $x_*$  renders  $g^*$  level sets invariant, it does not make  $\mathcal{G}$  attractive. Other control mechanisms that need not be feedforward might be used to move trajectories toward the manifold  $\mathcal{G}$ . To an investigator hoping to identify the control mechanisms of such a system, only the control rendering  $\mathcal{G}$

attractive would be apparent. The mechanisms generating the fast rejection of some perturbations encoded into the shape of  $\mathcal{G}$  itself might remain unexplained at best, or be mistakenly attributed to other mechanisms at worst.

The essence of our observation is not a surprising mathematical insight. Rather it is that elementary mathematical tools offer the possibility that one may encode a control law directly in the geometry of a hybrid transition. Doing so bypasses the need for conventional sensing and feedback calculations. This key observation we believe to be of potential use in the design of mechatronic systems and in the analysis of animal locomotion.

## 4.2 SLIP with Horizontal Velocity

As we noted above, the SLIP model is one of interest for examining dynamic running in both organisms and robots ([Full and Koditschek, 1999](#); [Blickhan, 1989b](#); [Seyfarth et al., 2006](#)). Using the results presented in §4.1.4, we consider the extension to the case with non-zero horizontal velocity.

The dynamics of the center of mass are extended to include an additional mechanical degree of freedom, increasing the number of physical states from two to four, with a horizontal absolute states of  $x$  and  $\dot{x}$ , and a vertical absolute states of  $y$  and  $\dot{y}$ . Again, the ground height is a piece-wise constant value per stride, with each stride being distinguished by an apex event of  $\dot{y} = 0$ .

In the aerial phases, the dynamics are given by ballistic motion:

$$\ddot{x} = 0$$

$$\ddot{y} = -g$$

In stance, the spring dynamics are the same Hill-like muscle model, where the force vector is co-linear with the leg. The massless leg has a rest length  $l_0$  which

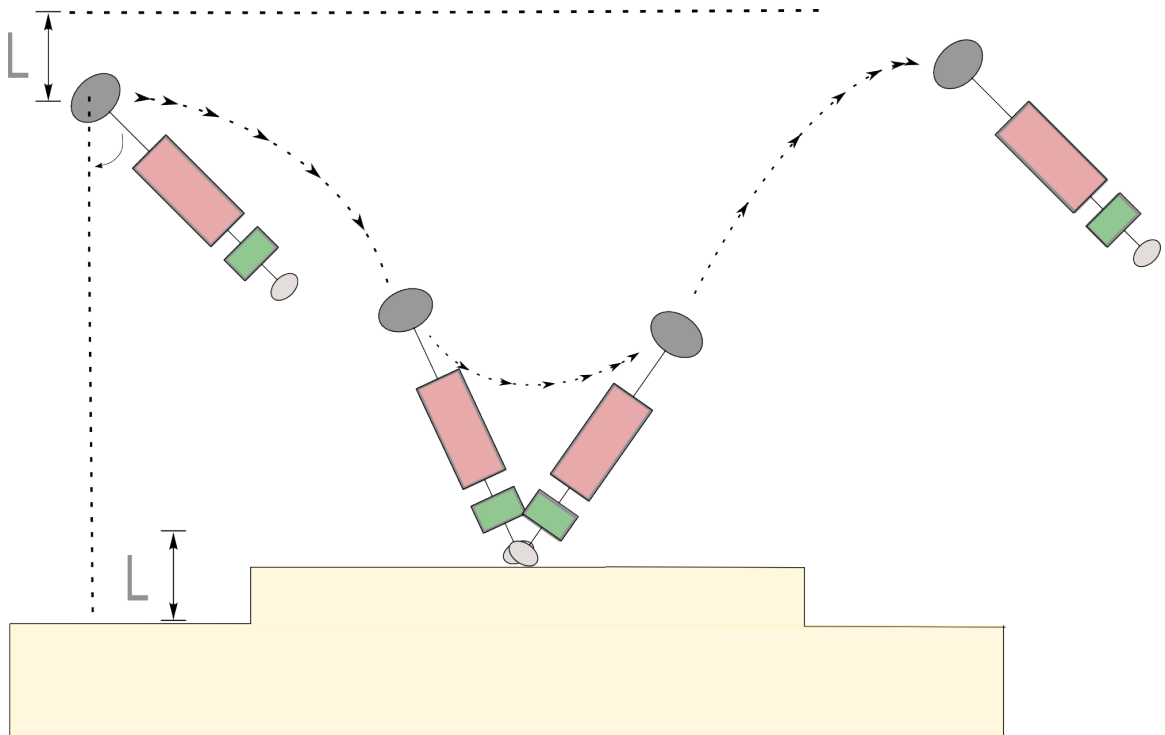


Figure 4.8: A depiction of an execution of a Spring Load Inverted Pendulum (SLIP) with horizontal velocity. The relative displacement above the ground, and horizontal velocity at apex are being preserved. The spring dynamics are non-conservative, so energy can be added or removed.



corresponds to zero compression, and a state-dependent length of  $l = \sqrt{x^2 + y^2}$  in general. The stance dynamics are :

$$\begin{aligned}\ddot{x} &= F_h \left( \frac{x}{l} \right) \\ \ddot{y} &= F_h \left( \frac{y}{l} \right) - g\end{aligned}$$

where  $F_h := K(L - l)(1 - \eta\dot{l}) - \mu\dot{l}$  for  $K, \eta, \mu$  playing the same role as previously. Thus, the leg dynamics are analogous (with an identical force law) to the vertical hopper, but the leg is non-vertical. The same qualitative behavior is investigated – only solutions which exhibit a distinct descent, stance, and ascent phase are of interest. We want to fix the relative apex height - given an unknown  $L$ , determine a strategy such that the relative height above the ground is preserved, i.e.

$$y_{apex}^i + L_i = y_{apex}^{i+1}$$

. We also add the requirement that the horizontal velocity  $\dot{x}$  is preserved at apex,

$$\dot{x}_{apex}^i = \dot{x}_{apex}^{i+1}$$

as we'd like to maintain the same horizontal speed.

#### 4.2.0.1 Simulation Environment

The system was simulated in Python 2.7.5 using the open-source numerical libraries NumPy and SciPy in concert with a python-port of the `dopri5` numerical integrator which supports event detection, identical to the environment used to simulate the vertical hopper above.

#### 4.2.0.2 Goal Function

In a departure from the vertical hopper, no explicit algebraic statement of a goal function  $g^*$  for the desired behavior is needed. The value of  $g^*$  was numerically evaluated. We did so by numerically integrating the stance dynamics until the impacted the terminal surface  $S$ , then evaluated  $g$ . We do this as (a) the SLIP dynamics are not integrable in closed-form ([Full and Koditschek, 1999](#)), and (b) illustrate that our method enables a “data-driven” approach; in Eqn. (4.2), we see that the flow  $\Phi$  only appears implicitly – on a trajectory, we can *directly* evaluate  $g$  on  $S$  without needing to compute the pullback.

A periodic solution for the system corresponds to a fixed point of the return map in body coordinates. As such, the exemplar  $x_*$  trajectory was determined by numerically solving for a fixed point of the return map for a ground height  $L = 0$ , whose nominal parameters are found in Table 4.2.

Table 4.2: Parameters

Parameter	Definition	Value
$\eta_0$	$F_V$ average slope	0.0826981
$\mu_0$	dissipative loss	1.93
$K$	$F_L$ average slope	100.0
$\alpha_0$	leg angle	$-40.86^\circ$
$y_{des}$	desired relative apex	1.2
$\dot{x}_{des}$	desired horizontal speed	5.0
$p_*(x, \dot{x}, y, \dot{y})$	known touchdown	(1.56, 5, 0.72, -3.08)
$l_0$	rest leg length	.937
$l_{to}$	toe-off height	0.0

Note that leg angle  $a_0$  is measured from a vertical normal to the ground. Positive angle would correspond to the COM in front of the foot, so the leg “extended” corresponds to a negative value.

#### 4.2.0.3 Feedforward Control Law

In [Seyfarth et al. \(2006\)](#), it was shown that a feed-forward angular acceleration of a leg's angle in similarly-configured SLIP model improved the stability properties of a periodic solution with respect to ground-height variation. Considering this, we alter our control scheme from full-state control to parametric control of the model parameters. Since the airborne robot has no means of effecting force on it's CoM, full-state control is not physically possible, so our motivation for parametric control is sound. During the descent phase, we restrict available control to parametric alteration of the spring parameters and the leg angle. Direct manipulation of the horizontal velocity or vertical velocity is not possible. We presume that the impact angle is perfectly controlled with no dynamics, so that  $\dot{\alpha} = 0$ ,  $\alpha(0) = \alpha_0 = u$ .

Instead of selecting a collection of initial states for the  $\mathcal{B}$  dynamics, and stipulating  $\Phi_{\mathcal{A}}$  coincide them in in the transition region, we will rather solve:  $\forall x \in \text{Im}(\Phi_{\mathcal{A}}) \cap \mathcal{O}$ , select a trajectory by varying  $\lambda$  from the family  $\Phi_{\mathcal{B}}(x, \lambda)$  such that  $g(x, \Phi_{\mathcal{B}}(x, T(x, \lambda))) = 0$ . We note this is possible as the vector fields are assumed Lipschitz in state, so that solutions vary smoothly with respect to parameters ([Khalil, 2002a](#)). Then, we define  $\hat{g}(x, \lambda) = g(x, \Phi_{\mathcal{B}}(x, T(x, \lambda)))$ . If  $D_{\lambda}\hat{g}$  has maximal rank, then via the Implicit Function Theorem, locally  $\exists p(x) \in \mathcal{C}^r$  s.t.  $\hat{g}(x, \lambda = p(x)) = 0$  ([Spivak, 1965](#)). In other words, there exists a function of compatible smoothness for each state to the parameter that permits the solution starting at that state to satisfy  $g$ . As a consequence, along the flow of  $f_{\mathcal{A}}$ , there is a function implicitly parameterized by time. The point being, there is no geometric distinction between controlling parameters versus states, since we are only interested in the map  $g$  itself. We can merely augment the state with our parameters, and compute the same quantities.

Implementation of our controller is, in abstract, determined by the time of a clock started at apex. As the mass descends, the parameters of the leg are reconfigured at each instant to provide the appropriate trajectory such that if the transition to

stance where to happen at that exact instant, the stance flow would satisfy  $g = 0$ . The outcome is entirely determined by the action taken in  $\mathcal{A}$ .

For our example, we considered the following case. Given a ground-height window of interest,  $\pm\delta L \in (-\epsilon + L_0, L_0 + \epsilon)$ , we discretized the region, and polynomially interpolated the resulting parameter sets as a function of  $\delta l$ . For each  $\delta L$  value, the python function minimizer method `fmin` was used to search over a parameter space – we arbitrarily selected the parameter space to be leg angle at impact  $\alpha$ , and the viscous damping coefficient  $\mu$ . The discrete point sets where then least-squared fit to a fourth order polynomial as a function of vertical velocity for each parameter, producing two control functions which are simultaneously applied.

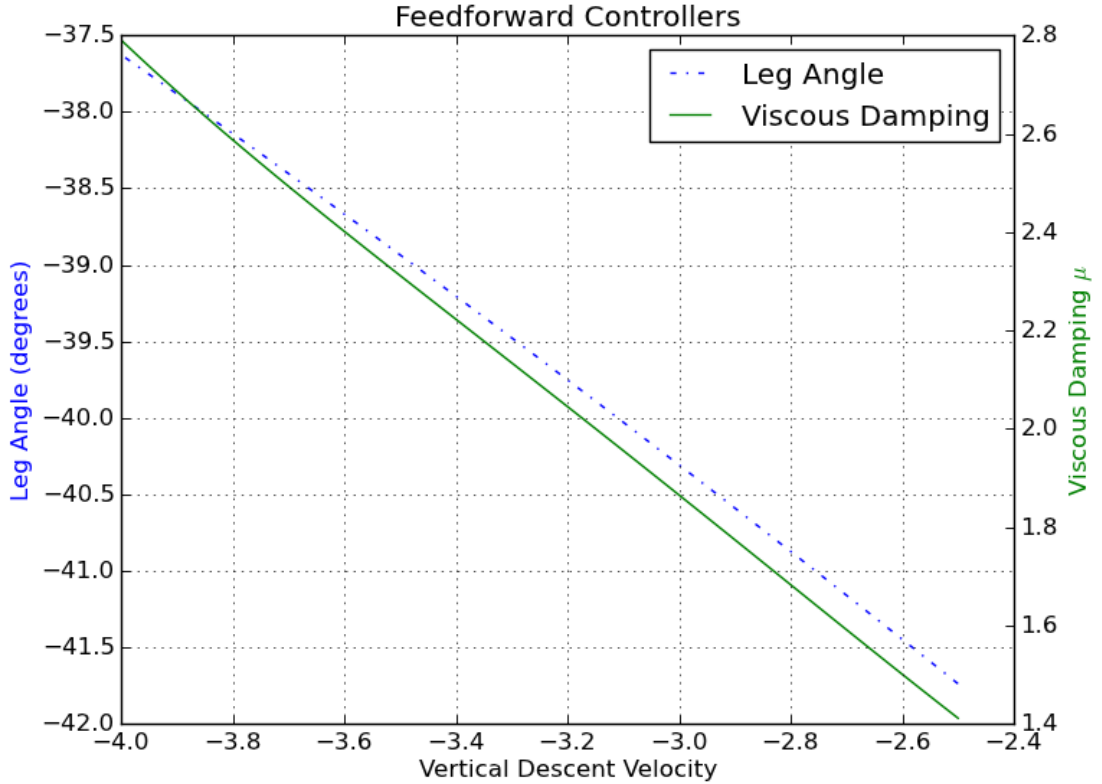


Figure 4.9: The feedforward control law during descent for varying the leg angle and damping as a function of the vertical velocity state, it is implicitly a function of time. The leg angle is consistent with [Seyfarth et al. \(2006\)](#), in that a *retraction* improves the stability properties.

Fig. 4.9 show the control laws generated for nominal gate parameters shown in Table 4.2.  $\delta L$  was varied between  $\pm 0.05$ , for a total variation of  $\sim 10\%$  the nominal leg length.

To verify performance, we selected a uniform random ground height sequence using the `numpy.random.rand`, over the specified range ( $\pm \delta L$ ). We expect the controller to fail for large  $\delta$ , as the implicit function theorem only provides *local* solutions. Figures 4.10 and 4.11 depict representative outcomes. The horizontal velocity and relative vertical position are preserved at 5 m/s and 1.2 m (Table 4.2) respectively at numerical precision, demonstrating deadbeat performance.

While Figures 4.10 and 4.11 are demonstrations of typical performance, for some randomly chosen sample paths, the controller would fail, and the SLIP would collapse (Fig. 4.12). The failure is likely due to the quality of the polynomial fit to the controller data. When we varied the order of the fit, it unpredictably improved or degraded performance, demonstrating that polynomials may not be a suitable set of basis functions for the generated curve, or the controller should be generated with a higher number of sample points. For example, increasing the number of  $\delta l$  samples by an order of magnitude was able to partially mitigate this effect, but not remove it completely. We acknowledge we possess an imperfect knowledge of how to discretize an embedded submanifold numerically. We argue the failures are a result of this knowledge gap, rather than a failure in our method's theory of existence. We admit intelligently solving for control inputs that can be reliably interpolated by basis functions may be challenging for systems that are strongly non-linear in their response to the transition state.

#### 4.2.1 Limitations

An issue observed with the leg-length controller seen for the vertical SLIP was that, for part of the transition region, the controller became non-physical in a naive

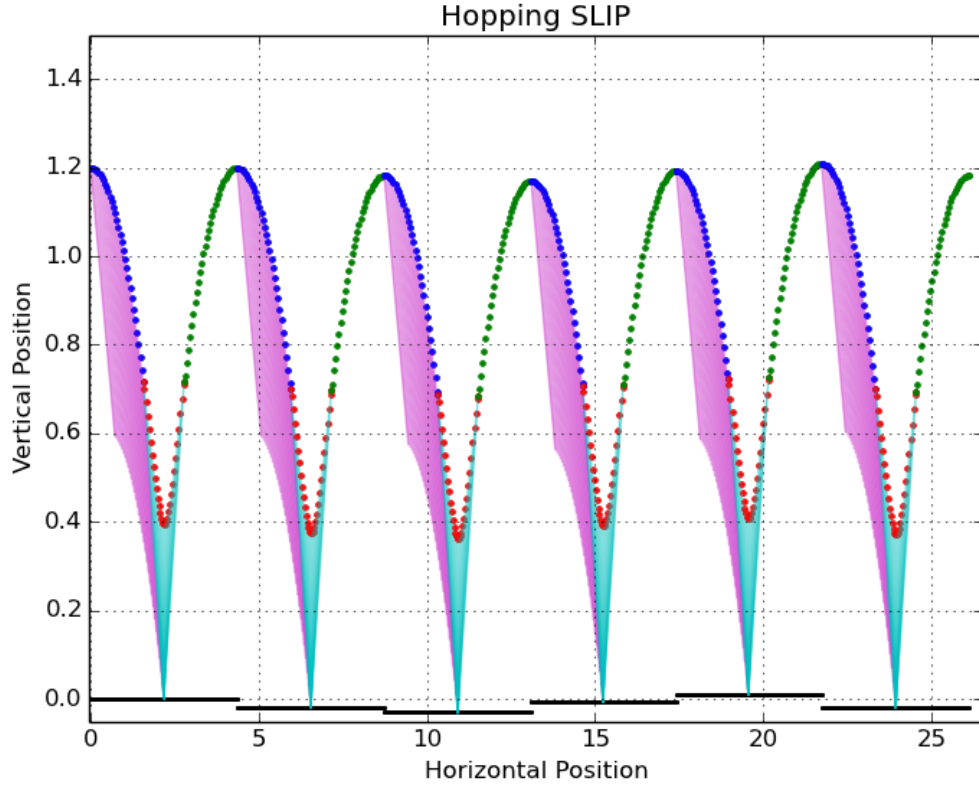


Figure 4.10: A seven-stride sample gait using the feedforward controller. Note that the axes are not equal, due to the large horizontal distance covered per-step compared to the vertical compression. Here, the pink line illustrates the current position of the leg during descent, while the cyan is the leg during stance. During descent, the leg angle and damping are governed by figure 4.9.  $\delta L = [0.0, -0.0179, -0.0302, -0.0071, 0.0093, -0.0190]$

sense, and would require potentially complicating factors in its design to be realizable. The controllers presented for the horizontal SLIP did not have such a difficulty, but would require a controllable mechanical impedance.

Our idea of tracking an invariant manifold is far from a silver bullet – it does not completely eliminate the need for control in some problems, and limitations on the class of rejectable disturbances were discovered attempting to extend it.

An issue observed with the leg-length controller seen for the vertical SLIP was that, for part of the transition region, the controller became non-physical in a naive sense, and would require potentially complicating factors in its design to be realizable.

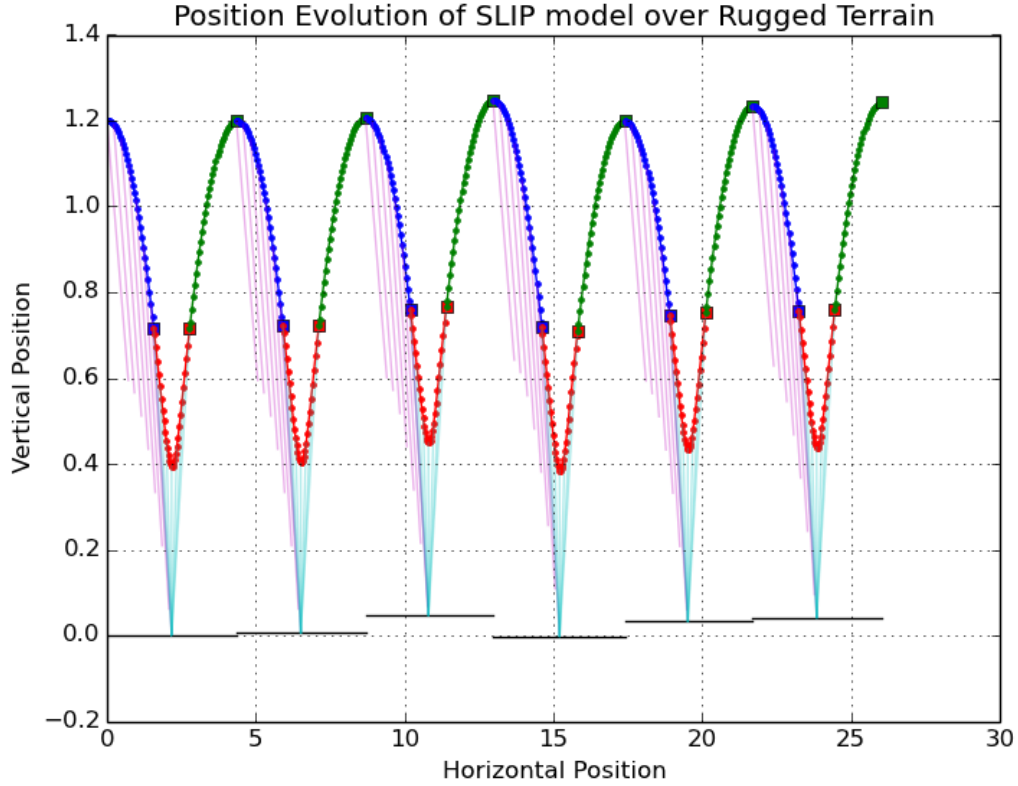


Figure 4.11: A seven stride gait under identical control as figure 4.10.  $\delta L = [0., 0.0060, 0.0473, -0.0018, 0.0332, 0.0417]$ . The leg is displayed less frequently compared to figure 4.10 for clarity - the controllers are still given by polynomials.

The controllers presented for the horizontal SLIP did not have such a difficulty, but would require a controllable mechanical impedance.

Another limitation is one that is a concern in robotics, where a problem archetype frequently encountered is applying a “controller” to a “plant”, i.e. finding signal  $u(t)$  so that the dynamics of the vector field  $f_{\mathcal{A}}$  exhibit a desired behavior. The  $\mathcal{G}$  manifold, our tracking objective, is not in general going to be an attractive surface, so an independent scheme would have to be used to create a suitable input signal to keep trajectories contained in this set. No general method is proposed here. The ease or difficulty of tracking the surface is determined by the dynamics and control freedom availability in  $\mathcal{A}$ , and requires separate analysis and design. Additionally, the level set is not defined outside of the transition region  $\mathcal{O}$ . It will likely be necessary

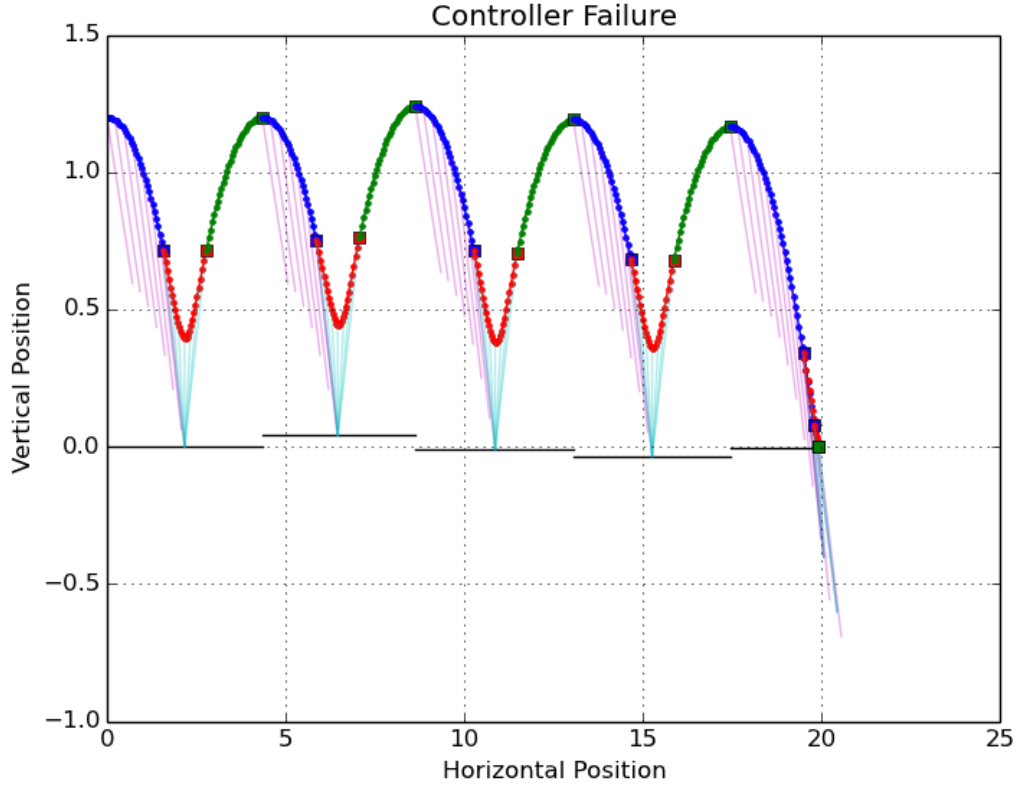


Figure 4.12: A gait where the controller has failed to stabilize the SLIP model with respect to ground height, illustrating that producing meaningful controller numerically may be computationally expensive to gain rich enough data sets for produce very reliable feedforward controllers.  $\delta l = ([0.0, 0.0408, -0.0090, -0.0345, -0.0024, 0.0414])$

to take action outside the neighborhood in  $\mathcal{O}$ , so that trajectories smoothly intersect and reside in  $\mathcal{G}$ .

An additional limitation that must be considered is the rank requirement that 0 is a regular value of  $g^*$ . While the rank of a smooth function is an open condition, it is clear this does not completely obviate the design problem of stipulating a full-rank cost function  $g$ . While the existence of submersions locally presents no obstacles, maintaining rank on an arbitrarily large set faces non-trivial topological obstructions ([Hirsch et al., 1974](#)).

Failure of rank can have physical significance, indicated there is a “real” problem,



as opposed to a badly chosen  $g$  that could be perturbed to have maximal rank; e.g., when attempting to extend our control scheme on the horizontal SLIP, we discovered that not all kinds of transition uncertainty are straightforward to resolve via the feedforward method we propose. When the uncertainty was in the ground stiffness, no feedforward controller could be found. We believe that this is an intrinsic limitation of the ground stiffness having no dependency on the chosen state variables. No motion in the air is capable of determining anything about the spring stiffness of the ground without contacting it. We could speculatively consider augmenting the state space through clever choice of auxiliary variables, but we did not pursue this idea further.

The control scheme associates with each point in  $\mathcal{O}$  an outcome, encoded in the goal function  $g$ . If the dependence of the uncertainty on state is degenerate, then there may not be a way to map a given transition state to a particular outcome. For example, given function  $T : \mathcal{O} \rightarrow R$  that maps out the transition events,  $T^{-1}(a)$  corresponds to those points in  $\mathcal{O}$  such that if  $\Phi_{\mathcal{A}}(t) \in T^{-1}(a)$ , it will transition into the uncontrolled region. The control method solves for a trajectory at each of those points that satisfies some control objective. If there is no way to parameterize those points as a function of the state in high control region  $\mathcal{A}$ , so that the flow may pass through them, it is not clear if a controller can be prescribed.

There are clearly general cases where such controllers would exist. For example, if the foliation  $G_a := \{x \in \mathcal{O} \mid G(x) = a\}$  is invariant (in the sense of e.g., [Lee \(2013, Prop 19.23\)](#)), then it is clear that a trajectory intersects each manifold once. Such a thing can be possible even for non-smooth flows, e.g. the event-selected vector fields of this document have this property with regards to their event functions  $h$ . However, if  $DT$  is not of maximal rank, then it is not immediately clear how to parameterize them.

Finally, there is a matter of implementability. For the horizontal SLIP example, the feedforward law is an action taken during the descent of the leg. The same motion

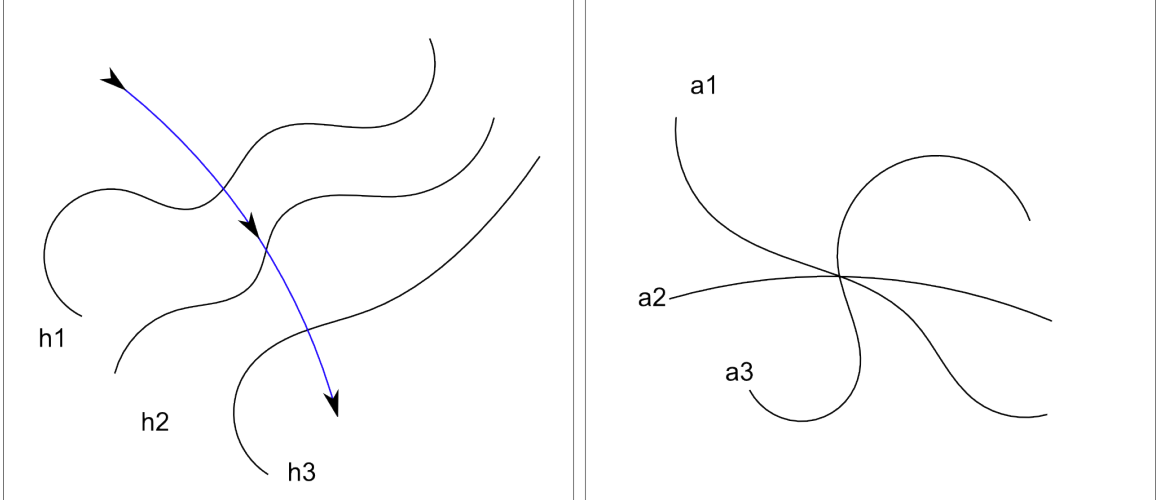


Figure 4.13: Examples of transition sets. The left figure shows a sequence of level sets, each a manifold, so there is a clear way to pass a trajectory through them. The red arc denotes a solution of the underlying vector field. The figure on the right shows a transition surface that is not an embedded submanifold. Here, the point of common intersection would pose a problem.

is repeated per stride, but during the ascent of the SLIP, the leg and damping will have to be reset to their starting values. Necessarily, the controller requires both accurate time-sensing, as well as accurate detection of the apex event. In general, practical models are going to need finite time to reset the system to prepare for the next execution.

#### 4.2.2 Implication For Mechanical Design

In natural environments, organisms share the problems robots encounter. The geometric controller presented here may provide a potential improvement to understanding how animals locomote, as an organism's performance is exceptional compared to most robots. Using the ideas presented here, an animal may be able to select a stride that mitigates the impact ground height uncertainty has on their gait.

Simultaneously, it compensates for the model uncertainty, as morphological features vary slightly across members of a species and even within the lifetime of an animal, yet they're able to maintain robust movement. A reasonable hypothesis is

that from the perspective of the animal, uncertainty in the environment is perceived similarly as morphological uncertainty of the body. The feedforward strategy prescribes a method to eliminate the impact of uncertain contact, be that due to an unknown ground height or an *unknown leg length*. If a limb is slightly longer or shorter than “nominal”, the same stride configuration can be used to preserve the gait of the animal. Such a scheme compensates for the minor variation present in living creatures without neural sensing.

Finally, while mentioned above, it bears repeating that the geometric control scheme imposes a dynamic constraint. A trajectory must remain within a level set of  $g^*$ ; while the problem was originally formulated to suggest that this would be a *tracking* problem, the factorization of the system into a “plant” and a “controller” is artificial. If the design problem is broadened to alteration of the “plant”, the unmodified dynamics of the system in  $\mathcal{A}$  could satisfy the control constraint without any input. Rather than design a controller to track the target submanifold, the plant could be designed so that its dynamics lay tangent to the submanifold already. This extends the control idea beyond the idea of formulating an external controller to force an undesirable plant to behave in a specific manner. Since the control problem is not dependent upon online data, it can be incorporated in the plant directly. The consequence of this observation is that the mechanical shape of the limb can directly encode the geometric controller, so that despite *no apparent feedback*, the gait would demonstrate perfect rejection of ground height variation. In the natural world, organisms could arrive at the proper limb morphologies through an evolutionary process – iteration over many generations could produce such a solution through trial-and-error. While the method proposes that animals have the *capacity* to achieve this kind of behavior, it remains an open question as to whether their structure exhibits these kind of features. From an engineering perspective, two similar considerations follow. Namely, that for some problems, intelligent design of the plant can satisfy a control

objective, and feedforward control methods could replace inner loop, high-bandwidth controllers. By embedding the control directly in the dynamics of the plant, disturbance rejection and model uncertainty insensitivity can be mitigated without external sensing.

### 4.3 Future Work

An eminently clear limitation of the suggested control strategy is the lack of stability of the feedforward trajectory, and the extension to multiple-contact systems. While it may be satisfactory to assume that proprioceptive feedback is adequate to stabilize the desired trajectory, another avenue for investigation is enabled by event selected vector fields. Motivated by the contractive “device” of 3.8, it may be possible to have totally mechanical stable deadbeat robot. Given a multi-limbed robot, one could use our feedforward technique to generate a deadbeat controller for the trajectory going the multi-contact point  $\rho$ . In 3.8, we saw how to stabilize general curves – presumably, one could engineer a “device” so that the feedforward trajectory is the stable attractor. Such a case would be interesting, as it could be done without *any* sensing. The mechanical structure could both naturally introduce event selected dynamics and deadbeat tracking exclusively through interaction with the ground. Such a mechanism would be highly resilient to variability in contact timing and sequencing, despite the lack of any apparent feedback mechanism.

## BIBLIOGRAPHY

### Bibliography

(2019), *ATOF Blade Fuses Rated 32*, LittleFuse.

Adolph, K. E., and S. R. Robinson (2013), The road to walking: What learning to walk tells us about development, *Oxford handbook of developmental psychology*, 1, 403–443.

Aizerman, M., and F. Gantmacher (1958), Determination of stability by linear approximation of a periodic solution of a system of differential equations with discontinuous right-hand sides, *The Quarterly Journal of Mechanics and Applied Mathematics*, 11(4), 385–398.

Alexander, K., et al. (1994), *Smooth invariant manifolds and normal forms*, vol. 7, World Scientific.

Alexander, R. M. (1984), The gaits of bipedal and quadrupedal animals, *The International Journal of Robotics Research*, 3(2), 49–59.

Ankarali, M., and U. Saranli (2010), Stride-tostride energy regulation for robust self-stability of torque-actuated dissipative spring-mass hopper, *Chaos*, 20(033121), doi:10.1063/1.3486803.

Arnold, V. (1973), *Ordinary differential equations*, The MIT Press, Cambridge, Massachusetts.

Arslan, O., and U. Saranli (2012), Reactive planning and control of planar spring-mass running on rough terrain, *IEEE Trans. Robotics*, 28, 567–579, doi:10.1109/TRO.2011.2178134.

Bellman, R. E. (1961), Dynamic programming treatment of the traveling salesman problem.

- Bernardo, M., C. Budd, A. R. Champneys, and P. Kowalczyk (2008), *Piecewise-smooth dynamical systems: theory and applications*, vol. 163, Springer Science & Business Media.
- Bizzarri, F., A. Brambilla, and G. S. Gajani (2013), Lyapunov exponents computation for hybrid neurons, *Journal of computational neuroscience*, *35*(2), 201–212.
- Blickhan, R. (1989a), The spring-mass model for running and hopping, *J. Biomechanics*, *22*(11-12), 1217–1227, doi:10.1016/0021-9290(89)90224-8.
- Blickhan, R. (1989b), The spring-mass model for running and hopping, *J. Biomechanics*, *22*(11-12), 1217–1227, doi:10.1016/0021-9290(89)90224-8.
- Bloch, A., J. Baillieul, P. Crouch, J. E. Marsden, D. Zenkov, P. S. Krishnaprasad, and R. M. Murray (2003), *Nonholonomic mechanics and control*, vol. 24, Springer.
- Bloch, A. M., P. Krishnaprasad, J. E. Marsden, and R. M. Murray (1996), Nonholonomic mechanical systems with symmetry, *Archive for Rational Mechanics and Analysis*, *136*(1), 21–99.
- Bogert, A., K. Gerritsen, and G. Cole (1998), Human musclemodelling from a user’s perspective, *J. Elecromyogr. Kines.*, *8*(2), 119–124, doi:10.1016/S1050-6411(97)00028-X.
- Bongard, J., V. Zykov, and H. Lipson (2006), Resilient machines through continuous self-modeling, *Science*, *314*(5802), doi:10.1126/science.1133687.
- Bongard, J. C. (2011), Morphological and environmental scaffolding synergize when evolving robot controllers: artificial life/robotics/evolvable hardware, in *Proceedings of the 13th annual conference on Genetic and evolutionary computation*, pp. 179–186, ACM.
- Browder, F. E. (1954), Covering spaces, fibre spaces, and local homeomorphisms, *Duke Math. J.*, *21*(2), 329–336, doi:10.1215/S0012-7094-54-02132-8.
- Bullo, F., A. D. Lewis, and K. M. Lynch (2002), Controllable kinematic reductions for mechanical systems: concepts, computational tools, and examples, in *Mathematical Theory of Networks and Systems*, vol. 124.
- Burden, S. A., S. Revzen, and S. S. Sastry (2015), Model reduction near periodic orbits of hybrid dynamical systems, *IEEE Transactions on Automatic Control*, *60*(10), 2626–2639.
- Burden, S. A., S. S. Sastry, D. E. Koditschek, and S. Revzen (2016), Event-selected vector field discontinuities yield piecewise-differentiable flows, *SIAM Journal on Applied Dynamical Systems*, *15*(2), 1227–1267.
- Carver, S., N. Cowan, and J. Guckenheimer (2009), Lateral stability of the spring-mass model suggests a two-step control strategy for running, *CHAOS*, *19*, doi:10.1063/1.3127577.

- Council, G., S. Yang, and S. Revzen (2014), Deadbeat control with (almost) no sensing in a hybrid model of legged locomotion, in *Advanced Mechatronic Systems (ICAMechS), 2014 International Conference on*, pp. 475–480, IEEE.
- Cully, A., J. Clune, D. Tarapore, and J. Mouret (2014), Robots that can adapt like animals, *Nature*, *521*(7553), doi:10.1038/nature14422.
- Da, X., O. Harib, R. Hartley, B. Griffin, and J. W. Grizzle (2016), From 2d design of underactuated bipedal gaits to 3d implementation: Walking with speed tracking, *IEEE Access*, *4*, 3469–3478.
- Deisenroth, M. P., G. Neumann, J. Peters, et al. (2013), A survey on policy search for robotics, *Foundations and Trends® in Robotics*, *2*(1–2), 1–142.
- Dieci, L., and L. Lopez (2011), Fundamental matrix solutions of piecewise smooth differential systems, *Mathematics and Computers in Simulation*, *81*(5), 932–953.
- Ernst, M., H. Geyer, and R. Blickhan (2011), Spring-legged locomotion on uneven ground: A control approach to keep the running speed constant, in *Int. Conf. on Climbing and Walking Robots*, pp. 639–644.
- Farley, C., R. Blickhan, and C. Taylor (1985), Mechanics of human hopping : model and experiments, *Am. Zool.*, *25*, 54A.
- Fenichel, N. (1974), Asymptotic stability with rate conditions for dynamical systems, *Bulletin of the American Mathematical Society*, *80*(2), 346–349.
- Fenichel, N. (1977), Asymptotic stability with rate conditions, ii, *Indiana University Mathematics Journal*, *26*(1), 81–93.
- Filippov, A. F. (2013), *Differential equations with discontinuous righthand sides: control systems*, vol. 18, Springer Science & Business Media.
- Full, R. J., and D. E. Koditschek (1999), Templates and anchors: neuromechanical hypotheses of legged locomotion on land, *Journal of experimental biology*, *202*(23), 3325–3332.
- Galewski, M., and M. Koniorczyk (2016), *Global invertibility theorems and their applications-a variational approach*, Wydawnictwo Politechnika Łódzka.
- Galloway, K. C., G. C. Haynes, B. D. Ilhan, A. M. Johnson, R. Knopf, G. Lynch, B. Plotnick, M. White, and D. E. Koditschek (2010), X-rhex: A highly mobile hexapedal robot for sensorimotor tasks, *Tech. rep.*, University of Pennsylvania.
- Ghigliazza, R., R. Altendorfer, P. Holmes, and D. Koditschek (2004), A simply stabilized running model, *SIAM J. App. Dyn. Systems*, *2*(2), 187–218, doi:10.1137/S1111111102408311.

- Ghigliazza, R. M., and P. Holmes (2005), Towards a neuromechanical model for insect locomotion: Hybrid dynamical systems, *Regular & Chaotic Dynamics*, 10(2), 193–225, doi:10.1070/RD2005v010n02ABEH000311.
- Goebel, R., R. Sanfelice, and A. Teel (2009), Hybrid dynamical systems, *Control Systems, IEEE*, 29(2), 28–93, doi:10.1109/MCS.2008.931718.
- Golubitsky, M., and V. Guillemin (2012), *Stable mappings and their singularities*, vol. 14, Springer Science & Business Media.
- Groff, R. E., P. P. Khargonekar, and D. E. Koditschek (2003), A local convergence proof for the minvar algorithm for computing continuous piecewise linear approximations, *SIAM journal on numerical analysis*, 41(3), 983–1007.
- Guckenheimer, J. (), Isochrons and phaseless sets, *Journal of Mathematical Biology*, 1(3), 259–273, doi:10.1007/BF01273747.
- Guckenheimer, J., and S. Johnson (1995), *Hybrid systems II*, 202–225 pp., Springer-Verlag, London, UK.
- Gupta, K., and A. P. Pobil (1998), *Practical motion planning in robotics: Current approaches and future directions*, John Wiley & Sons, Inc.
- Hairer, E., G. Wanner, and P. Norsett (1993), Solving ordinary differential equations i – nonstiff problems, Springer Series in Computational Mathematics, 2 ed., Springer-Verlag.
- Hatton, R. L., and H. Choset (2011), An introduction to geometric mechanics and differential geometry.
- Hatton, R. L., and H. Choset (2013), Geometric swimming at low and high reynolds numbers, *IEEE Transactions on Robotics*, 29(3), 615–624.
- Hereid, A., E. A. Cousineau, C. M. Hubicki, and A. D. Ames (2016), 3d dynamic walking with underactuated humanoid robots: A direct collocation framework for optimizing hybrid zero dynamics, in *Robotics and Automation (ICRA), 2016 IEEE International Conference on*, pp. 1447–1454, IEEE.
- Hill, A. (1938), The heat of shortening and the dynamic constants of muscle, *Proc. R. Soc. Lond. B*, 126(843), 136–195, doi:10.1098/rspb.1938.0050.
- Hirsch, M. W., R. L. Devaney, and S. Smale (1974), *Differential equations, dynamical systems, and linear algebra*, vol. 60, Academic press.
- Hirsch, M. W., C. C. Pugh, and M. Shub (2006), *Invariant manifolds*, vol. 583, Springer.
- Hirsch, M. W., S. Smale, and R. L. Devaney (2012), *Differential equations, dynamical systems, and an introduction to chaos*, Academic press.



- Holmes, P., R. Full, D. Koditschek, and J. Gukenheimer (2006a), The dynamics of legged locomotion : models, analyses, and challenges, *SIAM Review*, 48(2), 206–304, doi:10.1137/S0036144504445133.
- Holmes, P., R. Full, D. Koditschek, and J. Gukenheimer (2006b), The dynamics of legged locomotion : models, analyses, and challenges, *SIAM Review*, 48(2), 206–304, doi:10.1137/S0036144504445133.
- Hutter, M., R. C.D. M. Hoepflinger, and R. Seigwart (2010), Full state control of a slip model by touchdown detection, in *13th International Conference on Climbing and Walking Robots*, pp. 533–540.
- Ivanov, A. (1998), The stability of periodic solutions of discontinuous systems that intersect several surfaces of discontinuity, *Journal of Applied Mathematics and Mechanics*, 62(5), 677–685.
- Jarque-Bou, N., V. Gracia-Ibáñez, J.-L. Sancho-Bru, M. Vergara, A. Pérez-González, and F. Andrés (2016), Using kinematic reduction for studying grasping postures. an application to power and precision grasp of cylinders, *Applied ergonomics*, 56, 52–61.
- Jarque-Bou, N. J., A. Scano, M. Atzori, and H. Müller (2019), Kinematic synergies of hand grasps: a comprehensive study on a large publicly available dataset, *Journal of neuroengineering and rehabilitation*, 16(1), 63.
- Kelly, S. D., and R. M. Murray (1995), Geometric phases and robotic locomotion, *Journal of Robotic Systems*, 12(6), 417–431.
- Kenneally, G., A. De, and D. E. Koditschek (2016), Design principles for a family of direct-drive legged robots, *IEEE Robotics and Automation Letters*, 1(2), 900–907.
- Khalil, H. (2002a), *Non-Linear System: Third Edition*, Prentice Hall, Upper Saddle River, New Jersey.
- Khalil, H. (2002b), *Nonlinear systems*, 3 ed., Prentice Hall, Upper Saddle River, New Jersey.
- Kobayashi, S., and K. Nomizu (1963), *Foundations of differential geometry*, vol. 1, New York.
- Kober, J., J. A. Bagnell, and J. Peters (2013), Reinforcement learning in robotics: A survey, *The International Journal of Robotics Research*, 32(11), 1238–1274.
- Koditschek, D. E., and E. Rimon (1990), Robot navigation functions on manifolds with boundary, *Advances in applied mathematics*, 11(4), 412–442.
- Koon, W. S., and J. E. Marsden (1997), The geometric structure of nonholonomic mechanics, in *Proceedings of the 36th IEEE Conference on Decision and Control*, vol. 5, pp. 4856–4861, IEEE.

- Kuhn, H. W. (1960), Some combinatorial lemmas in topology, *IBM Journal of research and development*, 4(5), 518–524.
- Lee, J. M. (2013), Smooth manifolds, in *Introduction to Smooth Manifolds*, pp. 1–31, Springer.
- Levine, S., P. Pastor, A. Krizhevsky, J. Ibarz, and D. Quillen (2018), Learning hand-eye coordination for robotic grasping with deep learning and large-scale data collection, *The International Journal of Robotics Research*, 37(4-5), 421–436.
- Lygeros, J., K. H. Johansson, S. N. Simic, J. Zhang, and S. S. Sastry (2003), Dynamical properties of hybrid automata, *IEEE Transactions on automatic control*, 48(1), 2–17.
- Marsden, J. E., R. Montgomery, and T. S. Ratiu (1990), *Reduction, symmetry, and phases in mechanics*, vol. 436, American Mathematical Soc.
- Meigniez, G. (2002), Submersions, fibrations and bundles, *Transactions of the American Mathematical Society*, 354(9), 3771–3787.
- Mordatch, I., Z. Popović, and E. Todorov (2012a), Contact-invariant optimization for hand manipulation, in *Proceedings of the ACM SIGGRAPH/Eurographics symposium on computer animation*, pp. 137–144, Eurographics Association.
- Mordatch, I., E. Todorov, and Z. Popović (2012b), Discovery of complex behaviors through contact-invariant optimization, *ACM Transactions on Graphics (TOG)*, 31(4), 43.
- Müller, P. C. (1995), Calculation of lyapunov exponents for dynamic systems with discontinuities, *Chaos, Solitons & Fractals*, 5(9), 1671–1681.
- Murray, R., S. Sastry, and Z. Li (1994), *A mathematical introduction to robotic manipulation*, 1 ed., CRC Press.
- Murray, R. M. (2017), *A mathematical introduction to robotic manipulation*, CRC press.
- OLECH, C. (1998), On the wazewski equation, *ZESZYTY NAUKOWE-UNIwersytetu Jagiellonskiego-ALL SERIES-*, 1223, 55–64.
- Orhon, H. E. (2018), Model-based identification and control of a one-legged hopping robot, *arXiv preprint arXiv:1802.09634*.
- Ostrowski, J., and J. Burdick (1998), The geometric mechanics of undulatory robotic locomotion, *The international journal of robotics research*, 17(7), 683–701.
- Ostrowski, J. P. (1996), The mechanics and control of undulatory robotic locomotion, Ph.D. thesis, California Institute of Technology.

- Palmer III, L. R., and C. Eaton (2014), Periodic spring-mass running over uneven terrain using feedforward control of landing conditions, *Bioinspiration and Biomimetics*, 9(3), doi:10.1088/1748-3182/9/036018.
- Pepy, R., A. Lambert, and H. Mounier (2006), Path planning using a dynamic vehicle model, in *2006 2nd International Conference on Information & Communication Technologies*, vol. 1, pp. 781–786, IEEE.
- Pepyne, D. L., and C. G. Cassandras (2000), Optimal control of hybrid systems in manufacturing, *Proceedings of the IEEE*, 88(7), 1108–1123.
- Pratihari, D. K., K. Deb, and A. Ghosh (2002), Optimal path and gait generations simultaneously of a six-legged robot using a ga-fuzzy approach, *Robotics and Autonomous Systems*, 41(1), 1–20.
- Radford, J., and J. Burdick (1998), Local motion planning for nonholonomic control systems evolving on principal bundles, *A, A, 1*, 3.
- Ramezani, A., J. W. Hurst, K. A. Hamed, and J. W. Grizzle (2014), Performance analysis and feedback control of atrias, a three-dimensional bipedal robot, *Journal of Dynamic Systems, Measurement, and Control*, 136(2), 021,012.
- Ratliff, N., M. Zucker, J. A. Bagnell, and S. Srinivasa (2009), Chomp: Gradient optimization techniques for efficient motion planning, in *Robotics and Automation, 2009. ICRA '09. IEEE International Conference on*, pp. 489–494, IEEE.
- Reinkensmeyer, D. J., et al. (2014), Tools for understanding and optimizing robotic gait training.
- Reist, P., and R. D’Andrea (2012), Design and analysis of a blind juggling robot, *IEEE Transactions on Robotics*, 28(6), 1228–1243.
- Revzen, S., and J. Guckenheimer (2008), Estimating phase of synchronized oscillators, *Phys. Rev. E*, 78(5), 051,907, doi:10.1103/PhysRevE.78.051907.
- Revzen, S., and M. Kvalheim (2015), Data driven models of legged locomotion, in *SPIE Defense+ Security*, pp. 94,671V–94,671V, International Society for Optics and Photonics.
- Revzen, S., D. E. Koditschek, and R. J. Full (2009), Towards testable neuromechanical control architectures for running, in *Progress in Motor Control*, pp. 25–55, Springer.
- Revzen, S., M. Kvalheim, S. Wilshon, and J. Guckenheimer (2018), Estimating phase from observed trajectories, *In preparation*, pp. 1–20.
- Rockafellar, R. T. (2003), A property of piecewise smooth functions, *Computational Optimization and Applications*, 25(1-3), 247–250.
- Ruff, C. (2002), Variation in human body size and shape, *Annual Review of Anthropology*, 31, 211–232, doi:10.1146/annurev.anthro.31.040402.085407.

- Rumbaugh, J., I. Jacobson, and G. Booch (2004), *Unified modeling language reference manual, the*, Pearson Higher Education.
- Saranli, U., M. Buehler, and D. E. Koditschek (2001), Rhex: A simple and highly mobile hexapod robot, *The International Journal of Robotics Research*, 20(7), 616–631.
- Sastry, S. (1999), *Nonlinear systems : analysis, stability, and control*, Springer.
- Schmitt, J., and P. Holmes (2000), Mechanical models for insect locomotion: dynamics and stability in the horizontal plane i. theory, *Biological cybernetics*, 83(6), 501–515.
- Scholtes, S. (2012), *Introduction to piecewise differentiable equations*, Springer Science & Business Media.
- Seyfarth, A., H. Geyer, M. Gunther, and R. Blickhan (2002), A movement criterion for running, *J. Biomechanics*, 35, 649–655, doi:10.1016/S0021-9290(01)00245-7.
- Seyfarth, A., H. Geyer, and H. Herr (2006), Swing-leg retraction : a simple control model for stable running, *J. of Exp. Bio.*, 206, 2547–2555, doi:10.1242/jeb.00463.
- Sharir, M. (1989), Algorithmic motion planning in robotics, *Computer*, 22(3), 9–19.
- Siepel, J., and P. Holmes (2007), A simple model for clock-actuated legged locomotion, *Regular and Chaotic Dynamics*, 12(5), 502–520, doi:10.1134/S1560354707050048.
- Simić, S. N., K. H. Johansson, S. Sastry, and J. Lygeros (2000), Towards a geometric theory of hybrid systems, in *International Workshop on Hybrid Systems: Computation and Control*, pp. 421–436, Springer.
- Spivak, M. (1965), *Calculus on Manifolds*, Perseus Books Publishing, L. L. C.
- Spröwitz, A., A. Tuleu, M. Vespignani, M. Ajallooeian, E. Badri, and A. J. Ijspeert (2013), Towards dynamic trot gait locomotion: Design, control, and experiments with cheetah-cub, a compliant quadruped robot, *The International Journal of Robotics Research*, 32(8), 932–950.
- Tassa, Y., T. Erez, and E. Todorov (2012), Synthesis and stabilization of complex behaviors through online trajectory optimization, in *2012 IEEE/RSJ International Conference on Intelligent Robots and Systems*, pp. 4906–4913, IEEE.
- Tedrake, R., T. W. Zhang, and H. S. Seung (2004), Stochastic policy gradient reinforcement learning on a simple 3d biped, in *2004 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS)(IEEE Cat. No. 04CH37566)*, vol. 3, pp. 2849–2854, IEEE.
- Utkin, V. (1977), Variable structure systems with sliding modes, *IEEE Transactions on Automatic control*, 22(2), 212–222.

- Vejdani, H., Y. Blum, M. Daley, and J. Hurst (2013), Bio-inspired swing leg control for spring-mass robots running on ground with unexpected height disturbance, *Bioinspiration and Biomimetics*, 8(4), doi:10.1088/1748-3182/8/4/046006.
- Westervelt, E. R., J. W. Grizzle, and D. E. Koditschek (2003), Hybrid zero dynamics of planar biped walkers, *IEEE transactions on automatic control*, 48(1), 42–56.
- Winters, J. (1990), Hill-based muscle models: a systems engineering perspective, in *Multiple Muscle Systems*, edited by J. Winters and S.-Y. Woo, pp. 69–93, Springer New York, doi:10.1007/978-1-4613-9030-5\_5.
- Yosinski, J., J. Clune, D. Hidalgo, S. Nguyen, J. C. Zagal, and H. Lipson (2011), Evolving robot gaits in hardware: the hyperneat generative encoding vs. parameter optimization., in *ECAL*, pp. 890–897.
- Yu, H., M. Li, W. Guo, and H. Cai (2012), Stance control of the slip hopper with adjustable stiffness of leg spring, in *Mechatronics and Automation (ICMA), 2012 International Conference on*, pp. 2007–2012, IEEE.

## CHAPTER V

### Conclusion

What we have hoped to established with this document is that a substantial amount of structure can be attributed to legged robots without needing to fully define equations of motion, and that the structural form can be used to make strong predictions or prescriptions for stable gaits. In particular, the structures we have employed are accessible with data-driven techniques defining natural, simple, optimization problems that can be solved with shooting methods that presumably have better conditioned gradients compared to more conventional approaches. We have especially focused on methods to improve an exemplar motion – event-selected vector fields, behavioral specifications, and deadbeat hopping are all mutually attempting to improve the resiliency of a designer-specified behavior that is desired, but otherwise fragile. By anticipating that the desired motion is produced from a known class of models, but whose specific equations are unknown, additional structure can exploited to harden the motion to destructive disruption. In doing so, we hope to have convinced the reader that lessons from machine learning, theoretical mechanics, and control can be synthesized to achieve good outcomes. In other words, machine learning problems can be simplified with knowledge from mathematics, and conversely, analytically intractable objectives from theory can be re-cast as parsimonious optimization problems. By leveraging several perspectives simultaneously, we can exploit

their strengths, while mitigating some of their respective weaknesses.

## APPENDICES



## APPENDIX A

### Extension of $EC^r$ fields

Let  $\rho \in U \subset \mathbb{R}^d$  be a simply connected, precompact open set. Assume that  $h : U \rightarrow \mathbb{R}^m$  is uniformly continuous and thus bounded on  $U$ . We wish to investigate when it is possible to define  $f := (h, g)$  such that  $f$  is event-selected (and thus a diffeomorphism) with respect to the vector field  $F : U \rightarrow \mathbf{T}U$ .

Suppose that such an  $f := (h, g)$  existed, but  $Dg \cdot F$  was not sign-definite. Then, we can express

$$\begin{aligned} F(x) &= \sum_{i=1}^m \langle F(x), \nabla h_i(x) \rangle \cdot \nabla h_i(x) \\ &\quad + \sum_{k=m+1}^d \langle F(x), \nabla g_k(x) \rangle \cdot \nabla g_k(x) \end{aligned}$$

We know that  $\langle F(x), \nabla h_i(x) \rangle \geq c > 0$ . On any compact subset  $V$  of  $U$ ,  $\exists \alpha, \beta, \forall k, \langle \nabla g_k(x) \rangle \cdot \nabla g_k(x) \in (-\alpha, \beta)$ . We can define putative event functions

$$\hat{f}(x) := (h(x), 0, \dots, -\alpha g_k(x), \dots, 0)$$

, where the first  $m$  components are the original  $h$ , and the remaining  $d-m$  coordinates have  $-\alpha g_k(x)$  in the  $k$ -th place, and 0 in the rest. Then  $D\hat{f} \cdot F \geq c + \alpha > 0$ . Thus,

$\hat{f}$  is the desired diffeomorphism on  $V \subset U$ .

It is not immediately clear to the authors that defining such a  $g$  is an easy thing to do. We aim to give at least some conditions on which such a thing is possible.

### A.0.1 $f$ is proper

Suppose that  $f = (h, g)$  is a local diffeomorphism from  $U \rightarrow V \subset \mathbb{R}^n$ . To reiterate we insist that  $U$  is simply connected. If  $f$  is a covering map, then since  $U$  is simply connected, it must be diffeomorphic to any other covering map - e.g., it must be a diffeomorphism, as  $U$  is a universal cover. It is well-known that if  $f$  is a smooth local diffeomorphism that is also proper, then  $f$  is a smooth covering map. ([Lee, 2013](#), Prop 4.45) Thus, it is a diffeo on  $U$ .

### A.0.2 Path lifting

Roughly repeating the previous argument, if we can otherwise show that  $f$  is a covering map, it must be a diffeomorphism. One such condition is the path-lifting property ([Browder, 1954](#), Thm. 3). Let  $\gamma : [0, 1] \rightarrow \mathbb{R}^m$  be a rectifiable smooth path. We want  $\lambda(t)$  such that  $f(\lambda(t)) = \gamma(t)$ . If we differentiate, we get  $Df_{\lambda(t)} \cdot \lambda'(t) = \gamma'(t)$ , which allows us to write:  $\lambda'(t) = Df_{\lambda(t)}^{-1} \cdot \gamma'(t)$ . If  $\|Df_x \cdot v\| \geq \alpha \|v\|$ , equivalently,  $\|Df^{-1}\|^{-1} > b > 0$ .  $Df^{-1}$  is bounded and continuous, and since we demand  $\gamma'(t) \neq 0$ , this defines a complete differential equation with solution  $\lambda(t)$ . ([OLECH, 1998](#)) The global rank of  $\|DF\|$  can be employed very generally to conclude the existence of global diffeomorphisms by the Hadamard-Levy theorem ([Galewski and Koniorczyk, 2016](#), Thm. 3.12).

### A.0.3 $h$ defines a bundle

If the map  $h : U \rightarrow \mathbb{R}^m$  defines a fibre bundle over  $\mathbb{R}^m$ . A sufficient condition for this is that every set  $h^{-1}(x)$  is diffeomorphic to  $\mathbb{R}^{d-m}$  ([Meigniez, 2002](#), Cor. 31). By

the homotopy invariance of fiber bundles, and the fact that  $\mathbb{R}^m$  is contractible, the bundle is trivial. The global trivialization provides the requisite diffeomorphism.

#### **A.0.4 Sampled Vector Field**

If the piecewise constant vector field  $F$  has the event functions  $h_i := \langle e_i, \cdot \rangle$ , the previous examples demonstrate it is always possible.

## APPENDIX B

### Asymptotic Phase for Limit Cycles

Since our primary interest is in steady gaits, which we assume are defined by limit cycles, Eqn. (2.1) simplifies. Given an exponentially stable periodic solution, two points  $x, y$  are considered to be “asymptotically equivalent” if and only if, for open set  $D \subset \mathbb{R}$ , vector field  $f : D \rightarrow \mathbf{T}D$  with flow  $\phi(x, t)$ , and  $x, y \in D$

$$\lim_{t \rightarrow \infty} \|\phi(x, t) - \phi(y, t)\| = 0 \tag{B.1}$$

Denoting two such points via  $x \sim y$ , we observe that this is an equivalence relationship. As such, the equivalence classes partition  $D$ , and for each class, we denote a representative with  $\theta(x) \in D$ , and say  $\theta(x)$  is the “asymptotic phase” or simply “phase” of  $x$ .

For  $f \in \mathcal{C}^\infty$ , and an exponentially stable limit cycle for attractor, Winfree and Guckenheimer (*Guckenheimer*) illustrated that these equivalence classes (which they dub as “isochrons” as they are permuted under the flow  $\phi^t(x)$ ) have more structure. Denote the limit cycle of  $f$  by  $\Gamma \subset D$ . No two points of  $\Gamma$  are asymptotically equivalent, while every point in the stability basin  $B$  of  $\Gamma$  is equivalent to some point of  $\Gamma$ . Furthermore, each isochron is permuted under the flow, and are immersed

submanifolds that foliate the stability basin of the attractor. The following theorem (B.1) , which is a restatement of Theorem 3 of [Revzen and Kvalheim \(2015\)](#),

**Theorem B.1.** Asymptotic Phase and Isochrons

$\forall \beta \in \Gamma$ , let  $S_\beta := \theta^{-1}(\beta)$ . The isochron  $S_\beta$  is a  $C^\infty$  injective immersion of  $\mathbb{R}^{n-1}$ , and the map  $\theta : B \rightarrow \Gamma$  is a  $C^\infty$  submersion. Each isochron  $S_\beta$  is transverse to  $f$ , i.e.,  $\forall x \in S_\beta, f(x) \notin \mathbf{T}_x S_\beta$ . Additionally,  $B = \bigcup_{\beta \in \Gamma} S_\beta$ , and  $\phi(S_\theta(x), t) = S_{\theta(\phi(x, t))}$ .

*Remark* The map  $\theta$  has an image that is diffeomorphic to  $S^1$ , which allows phase to be naturally represented with parameterizations of the circle, so that the “phase of a point” is not a point on  $\Gamma$ , but instead a real or complex number in the chosen coordinates of  $S^1$ . In [Revzen et al. \(2018\)](#), phase can be equivalently defined with respect to an arbitrary point on  $\Gamma$ , leading to the definition of  $\tau : B \rightarrow [0, T)$ , where  $T$  is the period of  $\Gamma$ . While in this formulation, phase is a real number that denotes progress around the cycle from a reference point, it is not continuous at 0. The issue of “wrapping” around the circle continuously can be resolved by considering  $S^1 \subset \mathbb{C}$ , and defining  $\phi : B \rightarrow S^1$ , where  $\phi(x) := \exp(i\omega\tau(x))$ .

$\theta(x)$  can be used to define the *temporal 1 form* in the following fashion.  $\Gamma$  is one-dimensional. Therefore, there exists a unique  $\omega \in T^*\Gamma$  such that  $\omega(f|_\Gamma) = 1$ . Then, we have the pullback  $\theta^*\omega \in T^*\mathcal{B}$ . By definition,  $\theta^*\omega(f) = 1$  everywhere on  $\mathcal{B}$ .  $\omega$  is a 1-form (The closed form  $\omega$  is analogous gradient of asymptotic phase, but is not assumed to be exact) on  $B$  that allows phase comparison of vector fields through sign and magnitude.  $\omega(v) > 0$  would imply a curve is evolving in the same direction temporally as the isochrons;  $\omega(v) < 0$  would show the opposite, for example.

Geometrically, each isochron intersects the limit cycle once, the solutions starting on the same isochrons asymptotically coalesce together as time flows forward. Given that the limit cycle is stable, we know points in the stability collapse onto  $\Gamma$ , but the phase map identifies a specific point on  $\Gamma$  that points of the same phase evolve synchronously with, i.e, given a point  $x(0) \in S_{\theta(x)} \setminus \Gamma$ ,  $x(t)$  asymptotically collapses

to and flows synchronously with the solution starting at  $\theta(x) \in \Gamma$ .

The foliation of the stability basin into forward-commuting sets highlights that, for a given oscillator, states have a temporal ordering that is preserved *even off the limit cycle*. To have a specific periodic solution with desired dynamics, certain states come before other states, and through the perspective of recovery, suggests a recovery condition by preserving a sequence of motions. One posture follows another, etc.

The above discussion of phase is historical, and restricted to smooth systems. The alert reader will note that the models used in legged locomotion are *not* smooth - they are often hybrid systems with an aerial and ground domains, that are separated by a non-trivial reset map. While in each domain, the dynamics are smooth, the entire system is not.

The extension of asymptotic phase to hybrid systems is an open question. For classes of periodic hybrid system that have constant-rank Poincaré maps, Burden, Revzen, and Sastry ([Burden et al., 2015](#)) show that the hybrid dynamics can be smoothed by forming the adjunction space with a smooth structure for which the hybrid dynamics have a smooth periodic push-forward. The smooth system may admit the existence of asymptotic phase in the sense above, which could be lifted back to the original hybrid system. However, the lift may not itself be smooth - the isochrons may have cusps induced by each hybrid transition. As isochrons are extended to the entire stability basin of an oscillator by flowing backwards through time ([Guckenheimer](#)), a potentially infinite number of non-smooth points in each isochron develop as it approaches the boundary of the basin. Such a difficulty might be resolved through restricting to a compact neighborhood of the basin, for which only a finite number of degenerate points may exist. In even a more general periodic hybrid systems, the limit definition in 2.1 may still be considered, but the nature of the associated equivalence classes (isochrons) is less obvious, if they exist. The point being, the specific nature of an isochron in a hybrid system is still unclear, though several potential avenues for

generalization are present.